备受推崇、最超人气的
《鸟哥的Linux私房菜——服务器架设篇》
第三次改版升级



鸟哥的LINUX私房菜
——服务器架设篇（第三版）

鸟哥 著

鸟哥在本书中教您：
- 如何架设一个安全的网站
- 如何维护与管理您的服务器
- 如何有效地对您的服务器系统进行安全加固
- 如何在局域网与Internet中部署您的服务器

GOTOP

机械工业出版社
China Machine Press

目錄

鸟哥的Linux私房菜：服务器架设篇 第三版

> 来源：鸟哥的Linux私房菜：服务器架设篇

特别强调

底下这些文件主要是针对在 Linux 上的网络服务器来书写架设方式的，鸟哥主要以使用 RPM/YUM 作为软件安装的 CentOS 为基础系统。 CentOS 是属于 Red Hat Enterprise Linux (RHEL) 的操作系统，所以理论上 RHEL, CentOS, Fedora 等版本都适用的啦！ 为什么要使用默认的软件管理方式来安装所有的服务器程序呢？这是因为大多数的 Linux 开发商都会有所谓的在线升级系统， 包括 CentOS/Fedora 的 yum ，以及 SuSE 的 YOU ，还有 Debian 的 apt 等等， 因为有在线『自动升级』，所以当然会比您自己手动使用 Tarball 的安装方式来的方便且安全！ 因为你的系统上头所有的数据可以在第一时间内『自动』修补完毕嘛！

要架设好一部 Linux 服务器实在很简单，只要按照鸟哥网站上的步骤，一步一步的慢慢设定下去， 铁定可以完成您所需要的设定的！但是，要维护好一部 Linux 主机，却是很困难的！您必须要熟悉 Linux 的系统架构、网络的基本知识如协议、IP、路由、DNS 等等的基础知识才行！

无论如何，您要开始『服务器架设篇』之前，请务必先读完『Linux 基础篇』的文章才行！否则几乎就如同上面提到的，维护这样的一部 Linux 主机，是很花时间的！没有这个认知，那就不要架站吧！免得毒害其他的战友呐！

在架站的过程当中，无论出现任何问题，第一个步骤就是察看登录档 (log file)，那是克服问题的地方！

第一部份：架站前的进修专区

架站需要很强的 Linux 基础概念以及基础网络知识，否则的话，当网络断断续续的时候，您永远也不会知道是哪里出问题！ 而当某个服务器软件出问题的时候，您永远也不晓得是发生了什么事情！老人家说『对症下药才有效』， 随便吃药是不可能『无病强身』的！因此，对于网络服务器来说最重要的基础档案权限、程序之启动关闭与管理、 Bash shell 之操作与 script 、使用者账号的管理等等，您都必须要具备最基础的认知才行，否则，服务器真的不好碰！

在这一篇当中，鸟哥会介绍一下架设服务器之前你必须要具备的基础观念，以及重要的网络基础， 当然啦，一大堆的网络指令是需要熟悉的。这些网络指令不是要你背起来， 而是希望在你需要的时候可以很快速的查阅到如何使用的说！ 无论如何，请您务必在架站前『读过 Linux 基础篇』及『 读过网络基础篇』的文章，否则大家很难跟您讨论呢！ 这个部分鸟哥放在最前面，希望大家『务必』要查看这些资料啊！

作者序

最近更新日期：2011/08/18

作者序：

伺服器的架設並不容易，除了需要瞭解每個伺服器的工作原理與目的之外，還得要熟悉網路以及基礎系統管理操作等等。 不過目前有太多的書籍以及設定參考範例在教導大家如何架設一個可以用的伺服器，但這些範例卻沒有就伺服器的維護與管理， 還有發生問題時應該要如何處理的流程作個解釋。因此，架設伺服器是很容易的，不過，被攻擊也是很常見的啊！ 所以，筆者在這本書裡面就從系統基礎以及網路基礎講起，再談談網路攻擊後以及防火牆防護主機後，才進入架設伺服器的章節。

這本書是以 CentOS 6 爲範例來介紹的，這個版本的 Linux 有很多與以前不一樣的服務設定，常常會讓人找不到熟悉的設定檔位置。 而且筆者使用 SELinux 預設啓動的模式來進行伺服器的設定，加入 SELinux 後，整個伺服器的設定就顯的有些難度哩！ 此外，以前沒有用過的 NetworkManager 服務也來湊一腳，所以老是會讓人搞到滿腦子混亂～筆者光是重複測試之前版本與此版本的對應， 就花去不少的時間呢！希望這樣的測試結果，能夠幫大家降低自行試誤的過程，早點設定好您的伺服器。

這次第三版的改版幅度不算太大，主要是將前面幾章網路安全的部分文章統合，加入了第二版被拿掉的代理伺服器章節， 減少郵件伺服器的進階內容 (說實在的，郵件伺服器的架設眞的可以不用學太多了！)，並加入了相當重要的 iSCSI 這種磁碟提供者的模擬器！同時將 vsftpd 加入了 SSL 的加密支援！並且將伺服器常用在內部網路或網際網路作個區分， 以方便使用者瞭解該服務常用於哪些實務上。這些分類都是筆者近期來在學校作專題研究時的一些觀察後，所做的分析。 希望能夠對讀著們有些幫助。

誰適合這本書：

這本書既然是談論比較深入的架站規劃、流程、技巧與維護等工作，那麼比較基礎的 Linux 操作與相關的 Shell 語法，在這本書裡面就不可能談論的很多，畢竟， Linux 基礎篇 已經完成了，沒有必要在這本書裡面再次的重複提及的。 所以，當您嘗試閱讀這本書的時候，請注意，您最好已經具備有 Linux 作業系統的相關知識，以及文字介面 (BASH Shell) 的相關技巧，還有，必需要能夠瞭解一些 Unix-Like 的工作流程，例如登錄檔的產生與放置的地點、服

務的啓動與關閉方式、 工作排程的使用方法、以及其他種種相關的事項。也就是説，如果您從未接觸過 Linux ，那麼建議您由『鳥哥的 Linux 私房菜 -- 基礎學習篇』開始 Linux 的探索歷程，否則，這本書對您而言，可能會過於難以理解。

另外，這本書的內容很多時候會提到一些簡單的概念而不是僵化的流程， 尤其每個人對於網站的要求都不相同，也就是説，每個人的網站其實都是帶有個人風格的， 因此僵化的流程並沒有太大的意義～只要能夠依據這些簡單的概念來進行網站的架設，鳥哥認爲， 您的主機設定應該都不會有太大的問題。怕的是什麼呢？都沒有碰過 Linux ，卻想直接參考架站的程序來完成網站的架設的朋友，這些朋友最容易忽略後續的維護與管理了！ 這也容易造成網站的不穩定或者是造成被網路怪客 (Cracker) 入侵的問題啊！

這本書主要的目的是引導使用者進入 Linux 強大的網路功能的世界， 書內的範例都是鳥哥自己實際測試過沒有問題才寫上來的，不過，畢竟每個人的網路環境與操作習慣都不相同， 因此，鳥哥不敢説我書內的範例一定可以在您的系統上操作成功的！ 然而，書內都會提到一些基本概念的問題，只要理解這些基本的概念，並且對於 Linux 的操作熟悉，相信您一定可以利用書內的範例來開發出適合您自己的伺服器設定的！不過，對於沒有碰過 Linux 的朋友，還是建議從頭學起，至於爲什麼一定得從頭學起，在本書的第一章內會仔細談論喔。

章節安排：

本書在章節的規劃上面，主要分爲四大部分，分別是『網路基礎篇』、『主機的簡易防火措施篇』、『區網常見伺服器架設篇』 與『網際網路伺服器篇』，前兩篇的所有內容是很基礎的網路概念與實際網路設定，包括很重要的網路自我檢測以及防火牆設定等， 與您的伺服器能不能運作有很大的關係！所以，您在開始伺服器的架設之前，請務必將前面兩篇共十章先唸過一遍才好吶！

在『網路基礎篇』當中，我們會介紹簡易的網路基礎，這包含了硬體的選擇與佈線。此外，還有在 Linux 上面連上 Internet 的方法，以及在 Linux 發生無法連接網際網路的問題時，簡易的查驗方法。看完了這一篇之後，您的 Linux 不論以何種方式來進行 Internet 的連接，就應該都不成問題囉，而且，鳥哥希望看完這一篇之後，您可以瞭解 Linux 的網路問題，並自行解決喔！

在『主機的簡易防火措施篇』中，我們會簡單的介紹 Linux 的強大網路功能下，可能會發生的網路入侵問題。接下來，瞭解了問題後，當然就是需要來解決他囉！所以，我們會就 TCP/IP, port, 套件漏洞的修補與防火牆等來推敲一下，該如何做好 Linux 主機的防備呢？『沒有永遠安全的主機』是正確的言論，所以， 即使您的主機只是一個小小的網站，也千萬不能忽略這個防火牆的認識喔！

在『區網常見伺服器架設篇』當中，我們會介紹內部網路經常使用的遠端連線服務 (ssh, vnc, xrdp)、網路參數設定服務 (dhcp, ntp)、網路磁碟服務 (samba, nfs, iscsi)，以代理伺服器等服務。 這些章節雖然跳著看是沒有問題的，不過，鳥哥建議十一章的連線伺服器得要花些時間

瞧瞧，尤其是 ssh 的金鑰系統， 對於異地備援是相當有幫助的！

在『在網際網路伺服器篇』當中，我們會介紹 DNS, WWW, FTP 及 mail server 等常見的服務。 在網際網路上面要使用較好記的主機名稱來連線，就得要透過 DNS 系統，因此，這個 DNS 伺服器相當重要！在這一版的 DNS 加入了 view 的簡單概念，可以適用於區網內的主機聯網，可以參考看看。

章節的安排主要仍然是由淺入深來進行編排的，因此，還是希望讀者們可以由前面慢慢的往下看， 不要著急的直接翻到後面去抄一些架設流程喔！而且，幾乎每一章節後面都會具有一些簡單的課後練習題， 這些練習題有的是鳥哥參加過的考試內容，有的是鳥哥想到的一些資料，很適合大家思考喔！不要錯過這些練習題的訓練喔！

感謝：

感謝自由軟體社群志工們的軟體發展，讓我們能有這麼棒的作業系統來建置伺服器！也要感謝讀者們的回饋，讓鳥哥能夠在 Linux 伺服器的原理與設定方面有更深入的瞭解。感謝 Study Area 酷學園伙伴們的支持，包括 netman 大大、酷學園板主群、 鳥園討論板主群、以及參加實體活動的諸位朋友。感謝崑山科大資傳系張世熙主任與各位老師、伙伴們對不才小弟在研究方面的支援！ 更要感謝鳥哥的學生們，有你們的幫忙，讓鳥哥可以有較多的時間玩些伺服器測試與文章的撰寫！

最後，親愛的鳥嫂，謝謝你多年來的付出，尤其這兩年幫我們家添了兩個可愛的寶貝：宸宸與軒軒！ 希望鳥窩一家，以及所有的朋友們平安、幸福！

鳥哥 2011/08/18

另外，關於本書的戡誤資訊，請參考：

- http://linux.vbird.org/book/

2003/07/14：第一次完成日期！ 2003/09/18：加入一些說明，尤其是各個伺服器的簡介。a 2007/03/01：取消各個伺服器的簡介，太佔篇幅了！修改初版序的內容，增加再版序的部分。 2007/03/01：初版序我將他拿到 這裡 2011/08/18：二版序移動到這裡

第一章、架设服务器前的准备工作

最近更新日期：2011/07/14

很多朋友因为自身或服务单位的需求，总是有架设各种网络服务器的时刻，这个时候大多数的前辈都会推荐他们使用 Linux 做为服务器架设的操作系统。但因为这些朋友很多都没有受过 Linux 操作系统操作方面的训练，因此总觉得反正都是操作系统，所以 Linux 应该也跟 Windows 差不多吧！那么就硬着头皮使用图形接口去设定好众多的服务器，也有可能参考网络上一些文章， 即使是透过文字接口去设定，也能够很轻松的作好服务器的架设。问题是，这样的一部服务器是很容易被绑架的， 而且，如果网络不通，你如何自行将问题克服 (trouble shooting)？难道出问题只能无语问苍天？所以啰， 除非你只是暂时需要架设网络服务器，可以请朋友或其他信息公司帮你忙，如果你本身就是信息方面的服务提供商， 那鸟哥建议你在进行服务器实务设定之前，看一看这篇，试试看你到底有没有具备网络服务器的设定技能了呢？

1.1 前言： Linux 有啥功能

很多刚接触 Linux 的朋友常常会问的一句话就是：『我学 Linux 就是为了架设服务器，既然只是为了架设服务器，为什么我还要学习 Linux 的其他功能？例如：例行性工作排程、Bash Shell ，又干嘛去认识所有的登录档等等，我又用不到！此外，既然有好用的 Web 接口的 Server 设定软件，可以简单的将网站架设起来，为什么我还要去学习 vim 手动的编辑一些配置文件？干嘛还需要去理解服务器的工作的原理？』上面这些话对于刚刚学会架设网站的人来说，真是替他们道出了一个新手的心声啊！不过，对于任何一个曾经有过架设公开网站的朋友来说，上面这些话，真的是会害死人！为什么呢？底下我们就来分析一下。

1.1.1 只想用 Linux 架设服务器需要啥能力？

如果有人问你：『Linux 最强大的功能是什么』？大概大家都会回答『是网络功能啊！』，接下来，如果对方再问：『所以学 Linux 就是为了架设服务器啰？』呵呵！这个问题可就见仁见智啰！说穿了， Linux 其实就是一套非常稳定的操作系统，任何工作只要能在 Linux 这个操作系统上面跑，那他就是 Linux 可以达成的功能之一啰！所以 Linux 的作用实在不止于网络服务器的架设呐。

举例来说，在 Linux 上面开发跨平台的数值模式 (model) 诸如大型的大气仿真模式，由于 Linux 的稳定与完善的资源分配功能，使得在 Linux 上面开发出来的程序运作的又快又稳定。此外，诸如 KDE, GNOME 等漂亮的图形接口，搭配诸如 Open Office 等办公室软件，Linux 立刻摇身一变而成为优秀的办公室桌面计算机了 (Desktop)。此外，Google 制作出专门给手机系统用的 Android 也是以 Linux 为底开发的。所以说，千万不要小看了 Linux 的多样功能呐。

不过，不管怎么说， Linux 的强大网络功能确实是造成 Linux 能够在服务器领域内占有一席之地的重要项目。 既然如此，我们就好好的来探索一下 Linux 的网络世界吧！首先， Linux 到底可以达成哪些网络功能呢？这可就多着咯！不论是 WWW, Mail, FTP, DNS, 或者是 DHCP, NAT 与 Router 等等，Linux 系统都可以达到，而且，只要一部 Linux 就能够达到上面所有的功能了！当然，那是在不考虑网络安全与效能的情况下，你可以使用一部 Linux 主机来达成所有的网络功能。

但是你得要知道，『架站容易维护难』啊！更深一层来说，『维护还好、除错更难啊！』架设一个网站有什么难的？即使你完全没有摸过 Linux ，只要参考鸟哥的书籍或者是网站，而且一步一步照着做，包准你一个下午就可以架设完成五个以上的网络服务了！所以说， 架设服务器有什么难的？但要晓得的是，这样的一个网站，多则三天，少则数小时，立刻就会被入侵了！ 此外，被入侵之后，或许可以藉由一些工具来帮你将 root 的密码救回来，可惜的是， 这样的一个网站还是有被做为中继站的危险存在的！

另外，如果你使用工具 (例如 Webmin) 却怎么也架设不起来某个网站时，要怎么解决？如果你不懂该 Server 的运作原理与 Linux 系统的除错讯息，那么难道只能无语问苍天？不要怀疑这种情况的可能性， 参考一下各大论坛上面的留言就可以很清楚的知道这种情况的存在有越来越明显的趋势呢！

所以说，架设服务器之前还是有一些基本的技能需要学会的！而且这些技能是『一旦学会之后，眞正是终身受用啊！』只要花一个学期 (三~六个月) 就能学会一辈子可以使用的技能，这个学习的投资报酬率眞是太高了！ 所以，一开始的学习不要觉得苦，那眞的是值得的喔！^_^

Tips: 举例来说，鸟哥在 2003 ~ 2005 年跑去当兵了，当兵期间很少碰 Linux 啦！等到退伍后接到的第一个班要带 Linux 国际证照时，几乎所有的指令都在看不起鸟哥 @_@～不过，懂得学习的方法的鸟哥，透过 man 啦，透过 google 啦， 透过以前学习的一些概念啦，遇到问题几乎都可以在一分钟内解决，同学也不会有突然不知所云的困扰！你说， 这样是不是很好呢？



Linux 不是很好学，根据鸟哥过去教学的经验，很多同学在学 Linux 时眞是非常的痛苦，不过学完之后， 以前在 Windows 上面遇到的困难却也自然而然的迎刃而解！因为 Linux 训练我们时，是要我们去解决一个发现的问题， 这过程需要很多基础知识的培养，所以学完他之后，你会觉得很多事情都变的很简单而单纯。但如果使用 Windows 的懒人方案，很多问题就不可能了解为啥会发生与为啥可以这样处理了！我们会在下一节分析一下架设服务器的流程， 也会提供相对应的你应该要会的 Linux 技能喔！

1.1.2 架设服务器难不难呢？

不管是 Windows 还是 Linux ，要架设好一部堪称完美的服务器，『基本功课』还是得做的，这包括了：

1. 基础网络的基本概念，以方便进行联网与设定及除错；
2. 熟悉操作系统的简易操作：包括登录分析、账号管理、文书编辑器的使用等等的技巧；
3. 信息安全方面：包括防火墙与软件更新方面的相关知识等等；
4. 该服务器协议所需软件的基本安装、设定、除错等，才有办法实作。

而且，每一个项目里面所需要学习的技巧可多着呢！『什么？要学的东西那么多啊！』是啊！ 所以，不要以为信息管理人员整天闲闲没事干的呐，大家可是天天在出卖知识的，同时， 还得天天应付随时可能会发生的各种漏洞与网络攻击手法呢！眞不是人干的工作～～

这么说的话，架设服务器真的是挺难的喔！事实上，架设服务器其实蛮简单的哩！咦！～怎么又说架设服务器简单了？ 不是说架设服务器难吗？呵呵！其实『架设服务器很难』是由于朋友们学习的角度有点偏差的原因啦！ 还记得当初进入理工学院的时候，天天在念的东西是基础物理、基础化学、工程数学与流体力学等基础科目， 这些科目花了我们一至两学期的时间，而且内容还很难呐～都是一大堆的理论背不完。 怪了？我们进理工学院是为了求取更高深的知识，那么这些基础知识学了有什么用呐？ 呵呵！更高深的知识都是建构在这些基本科目的理论上面的，所以 万一你基础的科目没有读好，那么专业科目里面提到的基本理论怎么可能听的懂？

这样说应该就不难了解了吧！没错！认识操作系统与该操作系统的基本操作，还有那个重要的网络基础， 就是我们在架设服务器前的『基础科目』啦！所以说，在进入 Linux 的服务器世界之前，真的不能够略过网络基础的相关知识，同时， Linux 系统的基本技能也必需要能够理解呐！

好了，或许你还是对于 Linux 系统里面『什么是很重要的知识』不甚了解， 果真如此的话，那么我们就举个简单的例子来说明一下啰！底下列出一般的架设服务器流程， 我们由架设服务器的流程当中，来看一看什么是重要的 Linux 相关技能吧！ ^_^。

Tips: 在这一章当中，鸟哥不再就 linux 基础指令进行解析，因为在 『鸟哥的 Linux 私房菜 -- 基础学习篇』里面已经详细的介绍过了！ 如果持续的介绍指令，简直是浪费篇幅～所以底下仅介绍一个 Linux 基础学习重要性的分析喔！



1.2 基本架设服务器流程

虽然不同的服务器提供的服务并不相同，而且每种服务的原理也不见得都一样，不过，每种服务器由规划、架设到后续的安全维护， 其实整个流程是大同小异的。什么？你不相信啊？为了让你相信，那我们就来一项一项的分析看看吧！

1.2.1 网络服务器成功联机的分析

底下我们就整个服务器的简易架设流程当中来分析一下，以了解为什么了解操作系统的基础对于网站维护是相当重要的呢？首先，到底我们是如何联机到服务器的？ 联机到服务器又取得啥咚咚？我们先以底下这张图示来作个简单的说明好了：

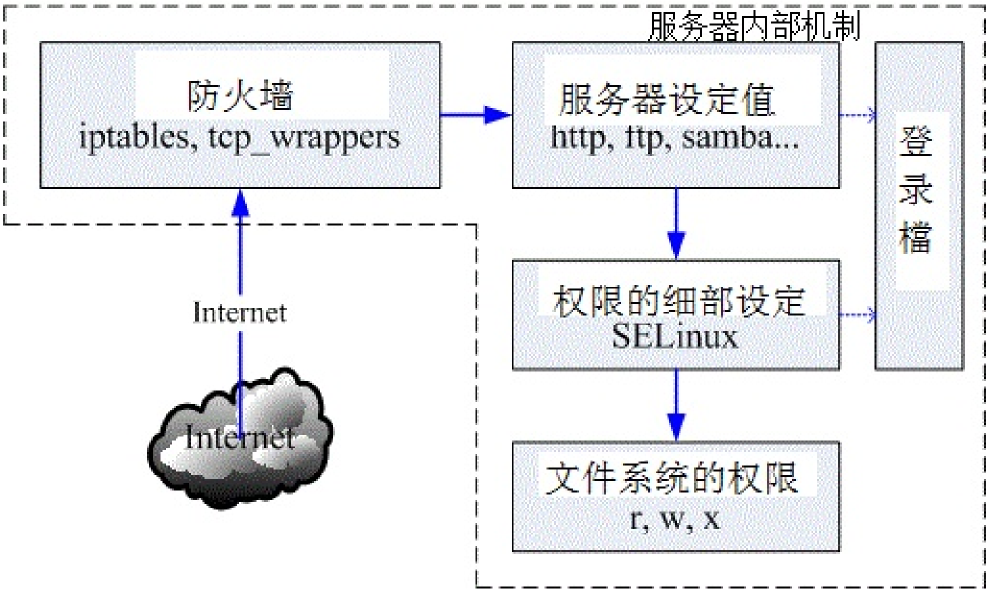


图 1.2-1、网络联机至服务器所需经过的各项环节

先来理解一下，到底我们联机到服务器想要得到什么？举例来说，你联机到 Youtube 想要看影片，所以对方就提供影片串流数据给你； 你连到 Yahoo 想要看新闻，所以对方就提供新闻的文本文件给你；你联机到无名小站想要看美女，对方就传图档给你；你联机 Facebook 想要去种田，对方就参考你之前留下来的记录，从数据库里面将你的记录拿出来传给你。看到没有，你联机到服务器，重点在取得对方的数据， 而一般数据的存在就是使用档案啰！那你有没有权限取得？最终与该文件系统的设定有关啦！

上图显示的是：首先，客户端到服务器的网络要能够通，等到客户端到达服务器后，会先由服务器的防火墙判断该联机能否放行， 等到放行之后才能使用到服务器软件的功能。而该功能又得要通过 SELinux 这个细部权限设定的项目后，才能够读取到文件系统。 但能不能读到文件系统呢？这又跟文件系统的权限 (rwx) 有关啦！上述的每个部分都要能够成功，否则就无法顺利读取数据啰。

所以，根据上面的流程我们大概可以将整个联机分为几个部分，包括：网络、服务器本身、内部防火墙软件设定、各项服务配置文件、细部权限的 SELinux 以及最终最重要的档案权限。底下就分几个细项来谈谈啰。

1. 网络：了解网络基础知识与所需服务之通讯协议

   既然要架设服务器，首先当然得要了解一下因特网。因为不管是哪种操作系统，若想要与因特网联机，这个网络基础就得了解。 举例来说，『网域』是经常会谈到的概念，当你发现一个设定为 192.168.1.0/255.255.255.0 时，晓得那是什么鬼东西吗？ 如果不知道的话，呵呵！绝对无法设定好网站的啦！另外，为何你需要服务器？当然是想要达成某项网络服务。 举例来说，传输档案可以用 FTP，那 WWW 可以传递档案吗？网芳可以传递吗？各有何用处？哪个比较方便？ 对于客户或老板来说，我们所设定的服务能否满足他们的需求等等，这都需要了解，否则你将一头雾水啊！ 因此这部份你就得要了解：

   - 基本的网络基础知识：包括以太网络硬件与协议、TCP/IP、网络联机所需参数等；
   - 各网络服务所对应的通讯协议原理，以及各通讯协议所需对应的软件。

2. 服务器本身：了解架网络服务器之目的以配合主机的安装规划

   想要架设服务器吗？那...架什么服务器？这个服务器要不要对 Internet 开放？这个服务要不要针对客户提供相关账号？ 要不要针对不同的客户账号进行例如磁盘容量、可活动空间与可用系统资源进行限制？如果要进行各项资源的限制， 那服务器操作系统应该要如何安装与设定？问题很多吧！所以，先了解你要的服务器服务目的之后，后续的规划才能陆续出炉。 不过，如果架站只是为了『练功』而已，呵呵！那就不需要考虑太多了～

3. 服务器本身：了解操作系统的基本操作

   网络服务软件是需要建置在操作系统上面的，所以基本的操作系统操作就得要了解才行啊！包括软件如何安装与移除？ 如何让系统进行例行的工作管理？如何依据服务器服务之目的规划文件系统？如何让文件系统具有未来扩充性 (LVM 之类)？ 系统如何管理各项服务之启动？系统的开机流程为何？系统出错时，该如何进行快速复原等等，这都需要了解的呢！

4. 内部防火墙设定：管理系统的可分享资源

   一部主机可以拥有多种服务器软件的运作，而很多 Linux distributions 出厂的默认值就已经开放很多服务给 Internet 使用了，不过这些服务可能并不是你想要开放的呢。我们在了解网络基础与所需服务的目的之后， 接下来就是透过防火墙来规范可以使用本服务器服务的用户，以让系统在使用上拥有较佳的控管情况。 此外，不管你的防火墙系统设定的再怎么严格，只要是你要开放的服务， 那防火墙对于该服务就没有保护的效果。因此，那个重要的在线更新软件机制就一定要定期进行！否则你的系统将会非常非常的不安全！

5. 服务器软件设定：学习设定技巧与开机是否自动执行

刚刚第一点就提到我们得要知道每种服务所能达成的功能，如此一来才能够架设你所需要的服务的网站。 那你所需要的服务是由哪个软件达成的？同一个服务可否有不同的软件？每种软件可以达成的目的是否相同？ 依据所需要的功能如何设定你的服务器软件？架设过程中如果出现错误，你该如何观察与除错？ 可否定期的分析服务器相关的登录信息，以方便了解该服务器的使用情况与错误发生的原因？ 能否通知多个用户进行联机测试，以取得较佳的服务器设定值？所以这里你可能就得要知道：

 - 软件如何安装、如何查询相关配置文件所在位置；
 - 服务器软件如何设定？
 - 服务器软件如何启动？如何设定自动开机启动？如何观察启动的埠口？
 - 服务器软件激活失败如何除错？如何观察登录档？如何透过登录档进行除错？
 - 透过客户端进行联机测试，如果失败该如何处理？联机失败的原因是服务器还是防火墙？
 - 服务器的设定修改是否有建立日志？登录档是否有定期分析？
 - 服务器所提供或分享的数据有无定期备份？如何定期自动备份或异地备份？

6. 细部权限设定：包括 SELinux 与档案权限

   等到你的服务器全部设定妥当，最后你所提供的档案数据权限却是给了『 000 』的权限分数， 那鸟哥很肯定的说，大家都无法读到你所提供的数据啊...！此外，新的 distributions 都建议你要启动 SELinux ，那是什么咚咚？ 如果你的数据放置于非正规的目录，那该如何处理 SELinux 的问题？又如何让档案具有保密性或共享性 (档案权限概念与 ACL 等) 等等，这也都是需要厘清的观念喔！

上述的服务器架设流程中，其实除了第 5 点之外，其他步骤在各服务器设定都需要了解啊！而且都是一样的东西说！ 因此，这些基础如果学会了，最终，你只要知道第 5 点里面那个软件的基础设定，你的服务器一下子就可以设定完成啦！ 这样说，你是否开始觉得基础学习很重要啊！ ^_^

1.2.2 一个常见的服务器设定案例分析

上面讲完后或许你还是不很清楚到底这些技能如何串起来？鸟哥这里提供一个简单的案例来分析一下好了， 这样你应该就比较容易清楚的知道为何需要学习这些咚咚。

- 网络环境：假设你的环境里面 (不管是家里还是宿舍) 共有五部计算机，这五部计算机需要串接在一起，且都可以对外联机；
- 对外网络：你的环境只有一个对外的联机方式，这里假设是台湾较流行的 ADSL 或 10M 的光纤这种透过电话线拨接的类型；
- 额外服务：你想要让这五部计算机都可以上网，而且其中还有一部可以做为网络驱动器机，提供同学或家人作为数据备份与分享之用；
- 服务器管理：由于你可能需要进行远程管理，因此你这部服务器得要开放联机机制，以让远程计算机可以联机到这部主机来进行维护；

- 防火墙管理：因为担心这部做为档案分享服务器的系统被攻击，因此你需要针对 IP 来源进行登入权力的控制；
- 账号管理：另外，由于同学的数据有隐密与共享之分，因此你还得要提供每个同学个别的账号， 且每个账号都有磁盘容量的使用限制；
- 后端分析：最后，由于担心系统出问题所以你得要让系统自动定期分析磁盘使用量、登录文件参数信息等等。

在上述的环境中，你要考虑的东西有哪些呢？依据本小节一开始谈到的六个步骤来分析的话，你可能需要底下这些咚咚喔！

1.2.2-1 了解网络基础

- 硬件规划

我们想要将五部计算机串接在一块，但是却又只有一个可以对外的联机，此时就得要购买集线器 (hub) 或者是交换器 (switch) 来串接所有的计算机了。但是这两者有何不同？为何 switch 比较贵？我们知道网络线被称为 RJ-45 的网络线， 但网络线材竟然有等级之分，这个等级要怎么分辨？不同等级的线材速度有没有差异？等到这些硬件基础了解之后， 你才能够针对你的环境来进行联机的设计。这部份我们等到下一章再来介绍。

- 联机规划

由于只有一条对外联机而已，因此通常我们就建议你可以用如下的方式来串接你的网络：

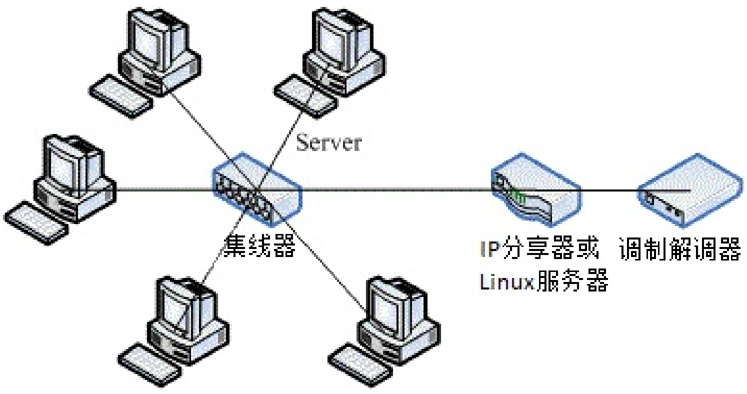


图 1.2-2、硬件的网络联机示意图

透过 IP 分享器，我们的五部计算机就都能够上网了。此时你得要注意，能否上网与 Internet 有关，Internet 就是那有名的 TCP/IP 通讯协议，而想要了解网络就得要知道啥是 OSI 七层协定。我们也知道能连上 Internet 与所谓的 IP 有关，那么我们内部这五部计算机所取得的 IP 能不能拿来架站？也就是说， IP 有没有不同种类？ 如果 IP 分享器突然挂了，那你的这五部计算机能不能联机玩魔兽？这就考虑你的网络参数设定问题了！

- 网络基础

如果你的同学或家人跑来跟你说，网络不通哩！你直觉会是什么？硬件问题？软件问题？还是啥莫名其妙的问题？ 如果你不懂网络基础的 IP 相关参数，包括路由设定以及领域名系统 (DNS) 的话，肯定不知道怎么进行联机测试的。 所以啰，此时你就会被骂说：『怎么都不懂还想要管理我们家网络』...那时不是很糗吗？所以要学好一些嘛！ 这部份就很复杂了，包括 TCP/IP, Network IP, Netmask IP, Broadcast IP, Gateway, DNS IP 等等，都需要理解喔！

了解了这些原理之后，你才能够进行除错 (debug) 的工作，否则，错误一出，你可能就会被骂的臭头的！ 最常见的错误中，举例来说，如果你的主机明明就可以使用 ping 这个指令去接触远方的主机 (ping IP)，但是就是无法使用 ping hostname 去接触远方的主机，请问，这个原因是什么呢？了解网络基础的朋友一看就知道几乎是 DNS 出问题了，不晓得的朋友就是想破头也得不到答案。既然知道出问题的地方，就能够针对该问题去处理嘛！

网络基础会影响到你的网络设定是否正确，这眞的很重要吶，因为，如果你的网络不通，那么即使服务器架设成功了， 别人可以看的到吗？所以说，要架站，眞的得对网络基础的部分下一些功夫才行的。关于网络基础这部份我们在基础篇并没有谈过， 所以我们会在下一章网络基础时再详加说明喔！

1.2.2-2 服务器本身的安装规划与架站目的的搭配

如同图 1.2-2 所示，Server 端是在那五部计算机之中，而且 Server 必须要提供针对不同账号给予网络驱动器机，我们这边会提供网芳 (SAMBA) 这个服务，因为他可以在 Linux/Windows 之间通用之故。 且由于需要提供账号给使用者，以及想到未来的磁盘扩充情况，因此我们想要将 /home 独立出来，且使用 LVM 这个管理模式， 并搭配 Quota 机制来控制每个账号的磁盘使用量。

所以说，你得知道 Linux 目录下的 FHS (Filesystem Hierarchy Standard) 的规范，否则分割槽给到错误的目录，会造成无法开机！那为什么要将 /home 独立放入一个分割槽？ 那是因为 quota 仅支持 filesystem 而不支持单一目录啊！好了，如果给你一部全新的主机，那你该如何安装你的系统呢？

实作题-全新安装：请到昆山科大 (http://ftp.ksu.edu.tw/FTP/CentOS/), 义守大学 (http://ftp.isu.edu.tw/pub/Linux/CentOS/) 或国家高速网络中心 (http://ftp.twaren.net/Linux/CentOS/) 下载最新的 Linux 映像档来刻录 (2011/07 可下载最新版为 CentOS 6.0)，并且依据上述的需求安装好你的 Linux 系统 (最重要的其实就是那个分割而已，其他的动作可以在安装完成后再说)。答：由于 Linux 的安装我们已经在基础篇内的第四章介绍过了，这里我们不再使用图形接口来说明， 仅使用文字说明来介绍你在每个项目应该处理的动作而已。此外，由读者们的响应发现，学习者经常只有一部主机， 因此，这里我们建议你使用 Virtualbox (http://www.virtualbox.org/) 来仿眞出一部实体主机，以安装你的测试环境。并请注意， 这部主机将会使用在本书的各个章节测试中。

Virtualbox 的安装与设定请自行参考其官网上面的 Documentation 介绍，这里不再赘言。只是需要注意的是， 若 (1)需要架设网站来上网，建议网络使用桥接模式 (bridge) ，且网络卡类型使用 Intel 的桌面计算机类型即可。 (2)由于我们未来会教导 NAT 服务器，因此最好有两张网卡，一张使用 bridge 一张使用内网 (intnet) 较佳。 而 (3)磁盘配置建议使用 SATA 类型，且容量请给予 25GB 以上。 (4)内存至少该给予 512MB 以上，最好有 1GB 来测试。 其他的请参考官网文件，或者使用默认配置即可。当然啦，如果你有独立的实体机器来安装，那就更好了！ 不需理会这一小段文字的说明喔。

默认配置如下：

- 分割表请依如下方式进行：
  - / + : 2GB
  - /boot: 200MB
  - /usr : 4GB
  - /var : 2GB
  - /tmp : 1GB
  - swap : 1GB
  - /home: 5GB，并且使用 LVM 模式建置
  - 其他容量请保留，未来再来进行额外练习！
- 软件挑选时，请选择『 basic server 』项目即可；
- 信息安全部分，防火墙选择启动，SELinux 选择强制 (Enforce)；
- 假设 IP 分享器有自动分配 IP 的功能，所以网络参数先选择 DHCP 即可，未来再自己修改。

实际流程大致如下 (鸟哥以 CentOS 6.0 为例说明)

1. 由于我们使用光驱开机来安装系统，因此得先进入 BIOS ，选择光驱开机，并且将 CentOS 6.x 的 DVD 放入光驱中；
2. 在启动安装的画面中，选择『Install or upgrade an existing system』来安装新系统；
3. 出现『 Disc Found 』字样，此时建议可以选择『 Skip 』即可略过；
4. 在欢迎画面以鼠标点选『 Next 』；
5. 语系数据可以选择『Chinese(Traditional)(中文(正体))』；
6. 键盘格式保留『美式英文』即可；
7. 安装包含的装置类型，直接选择默认的『基本储存装置』即可；
8. 因为我们是全新的硬盘，因此会出现一个找不到分割表的错误，此时选择『重新初始化』即可；
9. 进入网络主机名的设定，先保留『localhost.localdomain』即可。 同画面中还有一个『配置网络』的选项，我们先不要动他！等未来谈到网络设定再来处理即可；
10. 进入时区选择，请选择『亚洲/台北』即可；
11. 出现 root 密码制作，这里我们先设定为『 centos 』吧！ 这个密码太简单，系统会出现警告，你选择『照样使用』即可。你也可以自行设定其他密码；
12. 出现哪一类型安装的模式，因为我们有自己的分割考虑，所以，请选择『建立自定义分

割格式』来处理喔！

13. 在出现分割画面中，先点选『sda』项目，然后点选『建立』的按钮，在出现的窗口中，再点选『标准分割区』项目，然后点『建立』。 在最后的窗口中填写挂载点、容量等信息后，最终按下『确定』即可。最终画面有点像这样：



图 1.2-3、分割的参数下达示意图

14. 依据前面的分割规划，持续进行上述的动作，将所有的分割都处理完毕，除了 /home 之外。

15. 由于 /home 想要使用 LVM 的方式来建立文件系统，因此点选『建立』后，选择『LVM 实体卷册』项目，按下建立，在出现的分割窗口中容量填写 5GB，示意图有点像这样：



图 1.2-4、分割出 LVM 分割槽的方式

接下来回到原本的分割画面后，按下『建立』并选择『LVM 卷册群组』项目， 在出现的窗口中，卷册组名填写『server』，并且在右下方的逻辑卷册部分按下『新增』， 又会额外出现一个窗口，此时就填入 /home 的相关参数啦！注意，逻辑卷册我们这里设定为 myhome 喔！ 画面有点像底下这样：

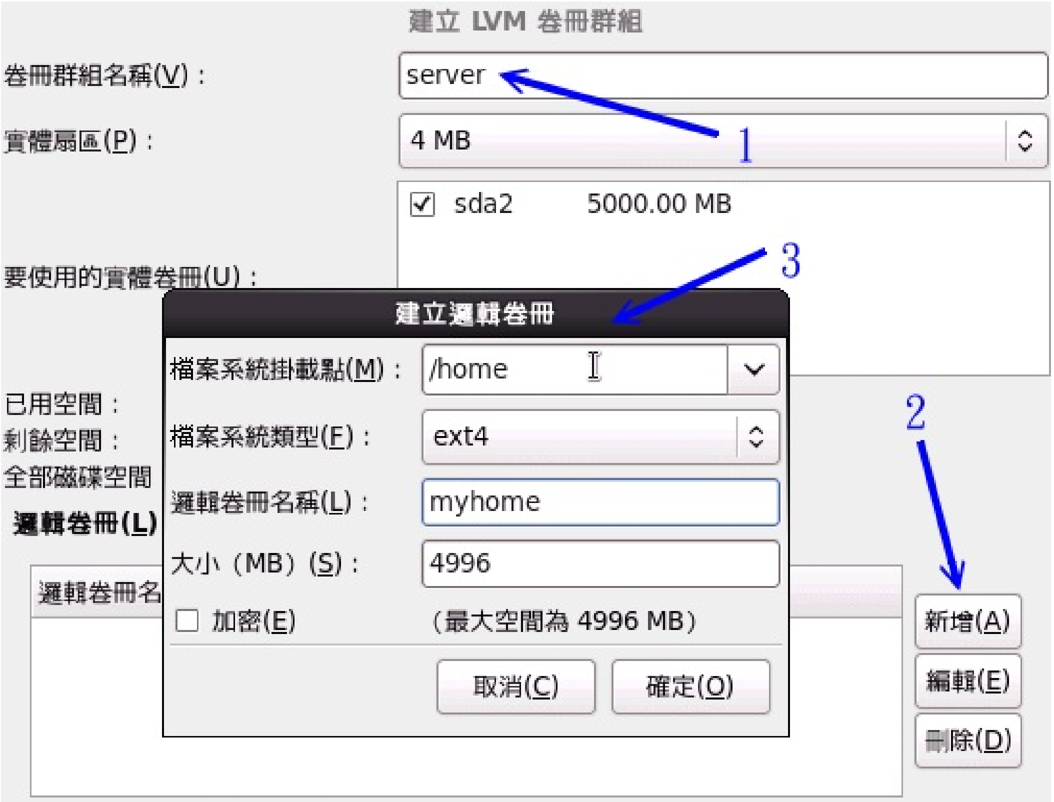


图 1.2-5、建立最终的 LVM 的 LV 与 /home

回到原本的分割画面，最终的显示有点像底下这样，然后请按下『下一步』继续。 但由于新建分割需要格式化，所以又会出现一个警告窗口！没问题的，选择『格式化』以及『将变更写至磁盘』吧！



图 1.2-6、分割的最终结果

16. 出现启动加载程序作业，都使用默认值即可，请按『下一步』；
17. 出现安装类型，因为我们主机的角色为服务器，因此选择『Basic Server』项目！ 其他项目保留默认，然后按下『下一步』就开始进行安装程序啰！
18. 经过一段时间的等待，出现重新启动后，你就重新启动吧！喔！要记得将 DVD 拿出来喔！(怪异的是，鸟哥第一次安装后， 竟然发现电源管理有问题，得在 kernel 处增加

noapic 才能顺利开机呢！)

19. 装好并重新启动后，就会进入 runlevel 3 的纯文本界面！因为是服务器嘛！

1.2.2-3 服务器本身的基本操作系统操作

既然我们这部主机得要提供不同账号来使用他们自己的网络驱动器，因此还需要建立账号啊，使用磁盘配额 (quota) 等等的。 那么你会不会建立账号呢？你会不会建置共享目录呢？你能不能处理每个账号的 Quota 配额呢？如果 /home 的容量不足了， 你会不会放大 /home 的容量呢？有没有办法将系统的磁盘使用情况定期的发送邮件给管理员呢？这些都是基本的维护行为喔！ 我们底下就以几个实际例子来练习看看你的基础能力吧！

例题-大量建置账号：假设我的五个朋友账号分别是 vbirduser{1,2,3,4,5}，且这五个朋友未来想要共享一个目录，因此应该要加入同一个群组，假设这个群组为 vbirdgroup，且这五个账号的密码均为 password 。那该如何建置这五个账号？答：你可以写一支脚本程序来进行上述的工作喔！

```
[root@localhost ~]# mkdir bin
[root@localhost ~]# cd /root/bin
[root@localhost bin]# vim useradd.sh
#!/bin/bash
groupadd vbirdgroup
for username in vbirduser1 vbirduser2 vbirduser3 vbirduser4 vbirduser5
do
    useradd -G vbirdgroup $username
    echo "password" &#124; passwd --stdin $username
done
[root@localhost bin]# sh useradd.sh
[root@localhost bin]# id vbirduser1
uid=501(vbirduser1) gid=502(vbirduser1) groups=502(vbirduser1),501(vbirdgroup)
context=root:system_r:unconfined_t:SystemLow-SystemHigh
```

最后利用 id 这个指令来查询看看，是否群组的支持是对的啊！

例题-共享目录的权限：这五个朋友的共享目录建置于 /home/vbirdgroup 这个目录，这个目录只能给这五个人使用，且每个人均可于该目录内进行任何动作！ 若有其他人则无法使用 (没有权限)，那该如何建置这个目录的权限呢？答：考虑到共享目录，因此目录需要有 SGID 的权限才行！否则个别群组数据会让这五个人彼此间无法修改对方的数据的。因此需要这样做：

```
[root@localhost ~]# mkdir /home/vbirdgroup
[root@localhost ~]# chgrp vbirdgroup /home/vbirdgroup
[root@localhost ~]# chmod 2770 /home/vbirdgroup
[root@localhost ~]# ll -d /home/vbirdgroup
drwxrws---. 2 root vbirdgroup 4096 2011-07-14 14:49 /home/vbirdgroup/
# 上面特殊字体的部分就是你需要注意的部分啰！特别注意那个权限的 s 功能喔！
```

例题-Quota 实作：假设这五个用户均需要进行磁盘配额限制，每个用户的配额为 2GB (hard) 以及 1.8GB (soft)，该如何处理？答：这一题实作比较难，因为必须要包括文件系统的支持、quota 数据文件建置、quota 启动、建立用户 quota 信息等过程。 整个过程在基础篇有讲过

了，这里很快速的带领大家进行一次吧！

```
# 1\. 启动 filesystem 的 Quota 支持
[root@localhost ~]# vim /etc/fstab
UUID=01acf085-69e5-4474-bbc6-dc366646b5c8 /      ext4  defaults  1 1
UUID=eb5986d8-2179-4952-bffd-eba31fb063ed /boot  ext4  defaults  1 2
/dev/mapper/server-myhome /home       ext4  defaults,usrquota,grpquota  1 2
UUID=605e815f-2740-4c0e-9ad9-14e069417226 /tmp   ext4  defaults  1 2
....(底下省略)....
# 因为是要处理用户的磁盘，所以找到的是 /home 这个目录来处理的啊！
# 另外，CentOS 6.x 以后，默认使用 UUID 的磁盘代号而非使用文件名。
# 不过，你还是能使用类似 /dev/sda1 之类的档名啦！
[root@localhost ~]# umount /home; mount -a
[root@localhost ~]# mount &#124; grep home
/dev/mapper/server-myhome on /home type ext4 (rw,usrquota,grpquota)
# 做完使用 mount 去检查一下 /home 所在的 filesystem 有没有上述的字眼！

# 2\. 制作 Quota 数据文件，并启动 Quota 支持
[root@localhost ~]# quotacheck -avug
quotacheck: Scanning /dev/mapper/server-myhome [/home] done
....(底下省略)....
# 会出现一些错误的警告信息，但那是正常的！出现上述的字样就对了！
[root@localhost ~]# quotaon -avug
/dev/mapper/server-myhome [/home]: group quotas turned on
/dev/mapper/server-myhome [/home]: user quotas turned on

# 3\. 制作 Quota 数据给用户
[root@localhost ~]# edquota -u vbirduser1
Disk quotas for user vbirduser1 (uid 500):
  Filesystem                 blocks    soft     hard    inodes  soft  hard
  /dev/mapper/server-myhome      20    1800000  2000000      5     0     0
# 因为 Quota 的单位是 KB ，所以这里要补上好多 0 啊！看的眼睛都花了！

[root@localhost ~]# edquota -p vbirduser1 vbirduser2
# 持续作几次，将 vbirduser{3,4,5} 通通补上去！

[root@localhost ~]# repquota -au
*** Report for user quotas on device /dev/mapper/server-myhome
Block grace time: 7days; Inode grace time: 7days
                        Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
root        --      24       0       0              3     0     0
vbirduser1 --      20 1800000 2000000              5     0     0
vbirduser2 --      20 1800000 2000000              5     0     0
vbirduser3 --      20 1800000 2000000              5     0     0
vbirduser4 --      20 1800000 2000000              5     0     0
vbirduser5 --      20 1800000 2000000              5     0     0
# 看到没？上述的结果就是有发现到设定的 Quota 值啰！整个流程就是这样！
```

例题-文件系统的放大 (LVM)：纯粹假设的，我们的 /home 不够用了，你想要将 /home 放大到 7GB 可不可行啊？答：因为当初就担心这个问题，所以 /home 已经是 LVM 的方式来管理了。此时我们要来瞧瞧 VG 够不够用，如果够用的话， 那就可以继续进行。如果不够用呢？我们就得要从 PV 着手啰！整个流程可以是这样来观察的。

```
# 1\. 先看看 VG 的量够不够用：
[root@localhost ~]# vgdisplay
  --- Volume group ---
  VG Name               server
  System ID
  Format                lvm2
....(中间省略)....
  VG Size               4.88 GiB  &lt;==只有区区 5G左右
  PE Size               4.00 MiB
  Total PE              1249
```

```
  Alloc PE / Size       1249 / 4.88 GiB
  Free  PE / Size       0 / 0     &lt;==完全没有剩余的容量了！
  VG UUID               SvAEou-2quf-Z1Tr-Wsdz-2UY8-Cmfm-Ni0Oaf
# 眞惨！已经没有多余的 VG 容量可以使用了！因此，我们得要增加 PV 才行。

# 2\. 开始制作出所需要的 partition 吧！作为 PV 用的！
[root@localhost ~]# fdisk /dev/sda  &lt;==详细流程我不写了！自己瞧

Command (m for help): p
   Device Boot      Start    End      Blocks   Id  System
....(中间省略)....
/dev/sda8            1812   1939     1024000   83  Linux &lt;==最后一个磁柱

Command (m for help): n
First cylinder (1173-3264, default 1173): 1940  &lt;==上面查到的号码加 1
Last cylinder, +cylinders or +size{K,M,G} (1940-3264, default 3264): +2G

Command (m for help): t
Partition number (1-9): 9
Hex code (type L to list codes): 8e

Command (m for help): p
   Device Boot  Start    End   Blocks  Id  System
/dev/sda9        1940   2201  2104515  8e  Linux LVM &lt;==得到 /dev/sda9

Command (m for help): w

[root@localhost ~]# partprobe &lt;==在虚拟机上面得要 reboot 才行！

# 3\. 将 /dev/sda9 加入 PV，并将该 PV 加入 server 这个 VG 吧
[root@localhost ~]# pvcreate /dev/sda9
[root@localhost ~]# vgextend server /dev/sda9
[root@localhost ~]# vgdisplay
....(前面省略)....
  VG Size               6.88 GiB       &lt;==这个 VG 最大就是 6.88G 啦
....(中间省略)....
  Free  PE / Size       513 / 2.00 GiB &lt;==有多出 2GB 的容量可用了！

# 4\. 准备加大 /home，开始前，还是先观察一下才增加 LV 容量较好！
[root@localhost ~]# lvdisplay
  --- Logical volume ---
  LV Name                /dev/server/myhome &lt;==这是 LV 的名字！
  VG Name                server
....(中间省略)....
  LV Size                4.88 GiB  &lt;==只有 5GB 左右，需要增加 2GB 啰
....(底下省略)....
# 看起来，是需要增加容量啰！我们使用 lvresize 来扩大容量吧！

[root@localhost ~]# lvresize -L 6.88G /dev/server/myhome
  Rounding up size to full physical extent 6.88 GiB
  Extending logical volume myhome to 6.88 GiB  &lt;==处理完毕啰！
  Logical volume myhome successfully resized
# 看来确实是扩大到 6.88GB 啰！开始处理文件系统吧！

# 5\. 扩大文件系统
[root@localhost ~]# resize2fs /dev/server/myhome
resize2fs 1.41.12 (17-May-2010)
Filesystem at /dev/server/myhome is mounted on /home; on-line resizing required
old desc_blocks = 1, new_desc_blocks = 1
Performing an on-line resize of /dev/server/myhome to 1804288 (4k) blocks.
The filesystem on /dev/server/myhome is now 1804288 blocks long.

[root@localhost ~]# df -h
文件系统              Size  Used Avail Use% 挂载点
/dev/mapper/server-myhome
                      6.8G  140M  6.4G   3% /home
....(其他省略)....
# 可以看到文件系统确实有放大到 6.8G 喔！这样了解了吗？
```

做完上面的实作之后，现在你晓得为什么在基础篇的时候，我们一直强调一些有的没有的了吧？因为那些东西在这里都用的上！ 如果本章这些题目你都不会，甚至连为什么要作这些东西都不懂的话，那得赶紧回去阅读基础篇，不要再念下去了！ 会非常非常辛苦的呦！

1.2.2-4 服务器内部的资源管理与防火墙规划

你可知道本章第一个实作题安装好了你的 Linux 之后，系统到底开放了多少服务呢？这些服务有没有对外面的世界开放监听？ 这些服务有没有漏洞或者是能不能进行网络在线更新？这些服务如果没有要用到，能不能关闭？此外， 这些服务能不能仅开放给部分的来源使用而不是对整个 Internet 开放？这都是需要了解的呢。 底下我们就以几个小案例来让你了解一下，到底哪些数据是你必须要熟悉的呢？

例题-不同 runlevel 的服务控管：在目前的 runlevel 之下，取得预设启动的服务有哪些呢？此外，我的系统目前不想启动自动网络挂载 (autofs) 机制，我不想要启动该服务的话，该如何处理？答：默认的 runlevel 可以使用 runlevel 这个指令来处理，那我们预设使用 3 号的 runlevel，因此你可以这样做：

```
[root@localhost ~]# LANG=C chkconfig --list &#124; grep '3:on'
```

上面指令的输出讯息中，会有 autofs 服务是在启动的状态，如果想要关闭他，可以这样做：

```
[root@localhost ~]# chkconfig autofs off
[root@localhost ~]# /etc/init.d/autofs stop
```

上面提到的仅只是有启动的服务，如果我想要了解到启动监听 TCP/UDP 封包的服务 (网络封包格式下章会谈到)，那该如何处理？ 可以参考底下这个练习题喔！

例题-查询启动在网络监听的服务我想要检查目前我这部主机启动在网络端口监听的服务有哪些，并且关闭不要的程序，该如何进行？答：网络监听的端口分析，可以使用如下的方式分析到：

```
[root@localhost ~]# netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address    Foreign Address   State  PID/Program name
tcp        0      0 0.0.0.0:111      0.0.0.0:*         LISTEN 1005/rpcbind
tcp        0      0 0.0.0.0:22       0.0.0.0:*         LISTEN 1224/sshd
tcp        0      0 127.0.0.1:25     0.0.0.0:*         LISTEN 1300/master
tcp        0      0 0.0.0.0:35363    0.0.0.0:*         LISTEN 1023/rpc.statd
tcp        0      0 :::111           :::*              LISTEN 1005/rpcbind
tcp        0      0 :::22            :::*              LISTEN 1224/sshd
tcp        0      0 ::1:25           :::*              LISTEN 1300/master
tcp        0      0 :::36985         :::*              LISTEN 1023/rpc.statd
udp        0      0 0.0.0.0:5353     0.0.0.0:*                1108/avahi-daemon:
udp        0      0 0.0.0.0:58474    0.0.0.0:*                1108/avahi-daemon:
....(底下省略)....
```

现在假设我想要关闭 avahi-daemon 这个服务以移除该服务启动的埠口时，应该要如同上题一样， 利用 /etc/init.d/xxx stop 关闭，再使用 chkconfig 去处理开机不启动的行为！不过，因为启动的服务名称与实际指令可能不一样， 我们在 netstat 上面看到的 program 项目是实际软件执行文件，可能与 /etc/init.d/ 底下的服务档名不同，因此可能需要使用 grep 去撷取数据，或者透过那好棒的 [tab] 按键去取得相关的服务档名才行。

```
[root@localhost ~]# /etc/init.d/avahi-daemon stop
[root@localhost ~]# chkconfig avahi-daemon off
```

我们常常会开玩笑说，如果对外开放的软件没有更新，那防火墙不过是个屁！所以啦，软件更新是相当重要的。 在 CentOS 内，我们已经有 yum 来进行在线更新了，你当然可以自己利用更改配置文件来指定 yum 要去查询的映像站 (mirror site)，不过这里鸟哥建议使用预设的设定值即可，因为系统会主动的判断较近的映像站 (虽然常常会误判)， 不需要人工微调啦！

例题-利用 yum 进行系统更新假设你的网络已经通了，目前你想要处理全系统更新，同时需要每天凌晨 2:15 自动进行全系统更新，该如何作？答：全系统更新使用 yum update 即可。但是由于 yum update 需要使用者手动输入 y 去确认真的要安装，因此在 crontab 里头处理相关任务时， 就得要使用 yum -y update 了！

```
[root@localhost ~]# yum -y update
# 第一次作会进行非常之久！因为系统真的有些数据要更新嘛！还是得等待的！

[root@localhost ~]# vim /etc/crontab
15 2 * * * root /usr/bin/yum -y update
```

不过这里还是要额外提醒各位喔，如果你的系统有更新过核心 (kernel) 这个软件，务必要重新启动啊！因为核心是在开机时载入的， 一经加载就无法在这次的操作中更改版本的。

那个 crontab 档案的处理，以及 crontab -e 的指令应用，内容的写法字段不太一样，请自行参考基础篇的说明去加强学习喔！

在通过了上述的各项设定后，我们的 Linux 系统应该是比较稳定些了，再接着下来，我们要开始来设定资源的保护了！ 例如 ssh 这个远程可登入的服务得要限制住可登入的 IP 来源，以及制订防火墙规则流程等。 这部份则是本教学文件后续要着重介绍的部分，留待后面章节再来谈吧！

Tips: 程序设计师所撰写的程序并非十全十美的，所以，总是可能有些地方没有设计好，因此就造成所谓的『程序漏洞』啰。 程序漏洞所造成的问题有大有小，小问题可能是造成主机的当机，大问题则可能造成主机的机密数据外流， 或者主机的操控权被 cracker 取得。在现今网络发达的年代，程序的漏洞问题是造成主机被攻击、入侵的最主要因素之一了。 因此，快速、有效的针对程序漏洞进行修补，是一个很重要的维护课题。



1.2.2-5 服务器软件设定：学习设定技巧与开机是否自动执行

这部份就是整个服务器架设篇的重要内容了！前一小节也曾谈过，在服务器架设部分你得要熟悉相当多的信息， 否则未来维护会显的很麻烦。我们以本章提到的大前提为例，我们想要提供一个网络驱动器机，那么网络驱动器机使用的机制有哪些呢？ 常见的除了网页形式的分享磁盘之外，还有常见的网芳以及 Linux 的 NFS 方式 (后面章节都会继续谈到)。

由于假设局域网络内的操作系统大部分是 Windows 好了，因此网芳应该是个比较合理的磁盘分享选择。 那么网芳到底启动了多少个埠口？是如何持续提供网芳数据的？提供的账号有没有限制？提供的权限该如何设定？ 是否可规定谁可登入某些特定目录？针对网芳服务的埠口该如何设定防火墙？如果系统出错该如何查询错误信息？ 这个网芳在 Linux 底下要使用什么服务来达成？这都是需要学习的呢！

直接告诉你，网芳的制作在 Linux 底下是由 Samba 这套软件来达成的。Samba 的详细设定我们会在后续章节介绍。 这里要告诉你的是， 架设一个网芳服务器，你应该要会的基础知识有哪些？以及告诉你，你可以背下来的架设流程中， 理论上应该要经过哪些步骤的过程，这样对你未来处理服务器设定时，才会有点帮助啊！

1. 软件安装与查询

   刚刚我们已经知道网芳需要安装的是 Samba 这套软件，那么该如何查询有没有安装呢？如果没有安装又该如何安装呢？ 那就来处理处理。

   例题：查出你的系统底下有没有 samba 这套软件，若无，请自行查询与安装该软件答：已安装的软件可以使用 rpm 去察看看，尚未安装的则使用 yum 功能。所以可以这样进行看看：

```
[root@localhost ~]# rpm -qa &#124; grep -i samba
samba-common-3.5.4-68.el6_0.2.x86_64
samba-client-3.5.4-68.el6_0.2.x86_64
samba-winbind-clients-3.5.4-68.el6_0.2.x86_64
# 看起来 samba 主程序尚未被安装啊！此时就要这样做：

[root@localhost ~]# yum search samba  &lt;==先查一下有没有相关的软件
[root@localhost ~]# yum install samba &lt;==找到之后，那就安装吧！

# 那么如何找出配置文件呢？因为我们总是需要修改配置文件啊！这样做吧：
[root@localhost ~]# rpm -qc samba samba-common
/etc/logrotate.d/samba
/etc/pam.d/samba
/etc/samba/smbusers
/etc/samba/lmhosts
/etc/samba/smb.conf
/etc/sysconfig/samba
```

2. 服务器主设定与相关设定

   这部份可就麻烦了！因为你得要了解到，你到底需要的服务是什么，针对该服务需要设定的项目有哪些？ 这些设定需要用到什么指令或配置文件等等。一般来说，你得要先察看这个服务的通讯协议是啥，然后了解该如何设定， 接下来编辑主配置文件，根据主配置文件的数据去执行相对应的指令来取得正确的环境设定。以我们这里的网芳为例， 我们需要设定工作组，然后需要设定可以使用网芳的身份为非匿名，接下来就能够开始处理主配置文件。 因此你需要有：

   i. 先使用 vim 去编辑 /etc/samba/smb.conf 配置文件；
   ii. 利用 useradd 建立所需要的网芳实体用户；
   iii. 利用 smbpasswd 建立可用网芳的实体帐户；
   iv. 利用 testparm 测试一下所有数据语法是否正确；
   v. 检查看看在网芳内分享的目录权限是否正确。 这些设定都搞定之后，才能够继续进行启动与观察的动作呦！而想要了解更多关于 samba 的相关设定技巧与应用， 除了 google 大神之外， /usr/share/doc 内的文件，以及 man 这个好用的家伙都必须要去阅读一番！

3. 服务器的启动与观察

   在设定妥当之后，接下来当然就是启动该服务器了。一般服务器的启动大多是使用 stand alone 的模式， 如果是比较少用的服务，如 telnet ，就比较有可能使用到 super daemon 的服务启动类型。我们这里依旧使用 samba 为例， 来瞧瞧如何启动他吧！

   例题：如何启动 samba 这个服务呢？并且设定好开机就启动他！答：想要了解如何启动，得要使用 rpm 去找一下软件的启动方式，然后再去处理启动的行为啰！

```
# 先查询一下启动的方式为何：
[root@localhost ~]# rpm -ql samba &#124; grep '/etc'
/etc/logrotate.d/samba
/etc/openldap/schema
/etc/openldap/schema/samba.schema
/etc/pam.d/samba
/etc/rc.d/init.d/nmb
/etc/rc.d/init.d/smb  &lt;==所以说是 stand alone 且档名为 smb, nmb 两个！
/etc/samba/smbusers

# 开始启动他！且设定开机就启动喔！：
[root@localhost ~]# /etc/init.d/smb start
[root@localhost ~]# /etc/init.d/nmb start
[root@localhost ~]# chkconfig smb on
[root@localhost ~]# chkconfig nmb on

# 接下来，让我们观察一下有没有启动相关的埠口吧！
[root@localhost ~]# netstat -tlunp &#124; grep '[sn]mbd'
tcp    0   0 :::139               :::*        LISTEN    1484/smbd
tcp    0   0 :::445               :::*        LISTEN    1484/smbd
udp    0   0 0.0.0.0:137          0.0.0.0:*             1492/nmbd
udp    0   0 0.0.0.0:138          0.0.0.0:*             1492/nmbd
```

最终我们可以看到启动的埠口有 137, 138, 139, 445 喔！

4. 客户端的联机测试

接下来就是要找一部机器做为客户端，然后尝试使用本机器提供的网芳功能啊！这样才能够了解设定是对还是错！ 相关的客户端联机与服务器提供的服务有关，例如 WWW 服务器就要使用 browser 去测试，网芳当然就得要使用网芳客户端程序啰！这部份也是本服务器篇要讲的基本内容啦！

但是很多时刻，客户端联机测试不成功并非是服务器设定的问题，很多是客户端使用方式不对！ 包括客户端自己的防火墙没开啦，客户端的账号权限密码等等记错啦等等的，问题很大啦！ 总体来说：『教育你的 Client 使用者具有最最基础的 Linux 账号、群组、档案权限等概念，才是一个彻底解决问题的方法』，但这也是最难的部分...

5. 错误克服与观察登录档

一般来说，如果 Linux 上面的服务出现问题时，通常会在屏幕上面直接告诉你错误的原因为何，所以你得要注意屏幕讯息。 老实说，屏幕讯息通常就已经告诉你该如何处理了。如果还不能处理呢？你可以这样处置看看：

- 先看看相关登录文件有没有错误讯息，举例来说， samba 除了会在 /var/log/messages 里面列出讯息外， 大部分的讯息应该是摆放在 /var/log/samba/ 这个目录下的数据，因此你就得先去查阅一番。通常在登录文件内的信息， 会比在屏幕上的还要仔细，那你就可以自行处理完毕了；
- 将讯息带入 Google 查询，通常可以解决登录档出现的但是你没有办法克服的问题喔！达标率可达 95% 以上吧！
- 还是不成功，那就到各大讨论区去发问吧！建议到酷学园 (http://phorum.study-area.org) 最常出现的其实是 SELinux 的错误啦！此时就得要使用 SELinux 的方法来尝试处理啰！ 这也是本服务器篇后续会稍微提到的内容。

经过上面的流程，你就可以知道啦，架设好一部主机需要知道：(1)各个 process 与 signal 的观念；(2)账号与群组的观念与相关性；(3)档案与目录的权限，这当然包含与账号相关的特性；(4)软件管理员的学习；(5)BASH 的语法与 shell scripts 的语法，还有那个很重要的 vim 啰！：(6)开机的流程分析，以及记录登录文件的设定与分析；(7)还得知道类似 quota 以及连结档等等的概念。要知道的真的很多，而且还是不能省略的步骤喔！

1.2.2-6 细部权限与 SELinux

如果有些特殊的使用情况时，权限设定就是个很重要的因素。举例来说，我们系统上面，现在有 vbirduser{1,2,3,4,5} 以及 student 等账号，而共享目录为 /home/vbirdgroup。现在，vbirdgroup 的群组想要让 student 这个用户可以进入该共享目录查阅， 但是不能够更改他们原本的数据，你该如何进行呢？你或许可以这样想：

- 让 student 加入 vbirdgroup 群组即可：但如此一来， student 具有 vbirdgroup 的 rwx 权限，也就可以写入与修改啰， 因此这个方案行不通。
- 将 /home/vbirdgroup 的权限改为 2775 即可：如此一来 student 拥有其他人的权限 (rx)，但如此一来其他所有任何人均拥有 rx 权限，这个方案也行不通。

传统的身份与权限概念就只有上面两种解决方案而已，这下子严重了！我们没有办法针对 student 进行权限设定！ 此时就得要使用 ACL 啰～同样这个例子，我们就来实作一下：

例题-单一用户、群组的权限设定 ACL想要让 student 可以进入 /home/vbirdgroup 进行查询，但不可写入。同时 vbirduser5 在 /home/vbirdgroup 内， 不具有任何权限。答：只能使用 ACL 啰！由于安装时预设格式化就加上 acl 的文件系统功能支持，因此你可以直接处理如下的各项指令。 如果你是使用后来新增的 partition 或 filesystem ，或许得要在 /etc/fstab 内额外增加 acl 控制参数才行喔！

```
[root@localhost ~]# useradd student
[root@localhost ~]# passwd student
[root@localhost ~]# setfacl -m u:student:rx /home/vbirdgroup
[root@localhost ~]# setfacl -m u:vbirduser5:- /home/vbirdgroup
[root@localhost ~]# getfacl /home/vbirdgroup
# file: home/vbirdgroup
# owner: root
# group: vbirdgroup
# flags: -s-
user::rwx
user:vbirduser5:---
user:student:r-x      &lt;==就是这两行，额外的权限参数哩！
group::rwx
mask::rwx
other::---

[root@localhost ~]# ll -d /home/vbirdgroup
drwxrws---+ 2 root vbirdgroup 4096 2011-07-14 14:49 /home/vbirdgroup
```

上面说的是正确的权限控制行为。那万一系统管理员不是个东西...不是啦！系统管理员并不知道权限的重要性时， 常常会因为某些特殊需求，就将整个目录设定为 777 的情况！举例来说，如果是一个不怎么想要负责的网管人员， 为了自己方便、大家方便，就将 /home/vbirdgroup 设定为 777 ，这样『大家欢喜』嘛！此时，如果你没有加上任何管理机制， 嘿嘿！这个群组成员工作的成果，通通可以被大家所窃取，真是要命了！

为了预防这种心不在焉的管理员，于是就有了 SELinux 这个玩意儿。SELinux 主要在控制细部的权限， 他可以针对某些程序要读取的档案来设计 SELinux 类别，当程序与档案的类别形态可以相符合时，该档案才能够开始被读取。 如此一来，当你配置文件案权限为 777 ，但是因为程序与档案的 SELinux 例行不符，所以没关系的，因为该程序还是读不到该档案！ 所以我们在图 1.2-1 才会将 SELinux 的图示绘制到 daemon 与 file permission 中间啊！

事实上 SELinux 还挺复杂的，但是我们如果仅是想要应用而已，那么 SELinux 的处理方式通通可以透过登录档来处置！ 所以 SELinux 出现问题的机会非常大，但是解决技巧却很简单！就是透过登录档内的说明去作即可。 详细的作法我们在后续章节再持续说明吧！

1.2.3 系统安全与备份处理

老实说，在鸟哥管理服务器的经验来说，硬件问题要比操作系统与软件问题还来的严重，而人的问题又比硬件问题严重！ 举例来说，如果你的老板跟你说：『我要的账号是 eric ，而且我的密码也要是 eric ！这样比较好记嘛！』 你应该要怎么处理呢？『果然需要再教育』！教育谁？教育自己啦！是要忍耐还是要说服老板别这样～好讨厌的感觉吧！

因此，在系统安全方面，首要的工作是透过日常生活的社交活动中，慢慢透露一些资安方面的困扰， 并提供老板一些制订资安规则方面的信息，这样未来比较好鼓吹资安条件的制订。 我们就先由严格的密码来建议吧：

『猜密码』仍是一个不可忽视的入侵手段！例如 SSH 如果对 Internet 开放的话，你又没有将 root 的登入权限关闭，那么对方将可能以 root 尝试登入你的 Linux 主机，这个时候对方最重要的步骤就是猜出你 root 的密码了！如果你 root 的密码设定成『1234567』哈哈！想不被入侵都很难～ 所以当然需要严格的规范用户密码的设定了！那么如何规范严格的密码规则呢？ 可以藉由 (1)修改 /etc/login.defs 档案里面的规则，以让用户需要每半年更改一次密码，且密码长度需要长于 8 个字符呢！(2)利用 /etc/security/limits.conf 来规范每个使用者的相关权限，让你的 Linux 可以较为安全一点点～(3)利用 pam 模块来额外的进行密码的验证工作。

另外，虽然『防火墙无用论』常常被提及，但是 netfilter (Linux 的核心内建防火墙) 其实仍有他存在的必要。 因此你还是得就要你自己的主机环境来设计专属于自己的防火墙规则，例如上面提到的 SSH 服务中， 你可以仅针对某个局域网络或某个特定 IP 开放联机功能即可啊！

最后，备份是不可忽略的一环。本节开头就讲到了，鸟哥遇过常常莫名其妙自动重开机或系统不稳的，经常都不是被攻击， 而是硬件内部的电子零件老化所造成的系统不稳定...此时，异地备援啦、备用机器的接管理等等的，就很重要啰！ 而你总不想要因为硬盘挂点导致数据

『害害去』，所以啰，备份就真他X的重要啰！

例题：系统上比较重要的目录有 /etc, /home, /root, /var/spool/mail 等，你现在想要在每天 2:45am 进行备份，且备份数据存到 /backup 内， 备份的举动使用 tar ，那该如何处理？答：鸟哥通常是使用 shell script 来进行备份数据的汇整，范例如下：

```
[root@localhost ~]# mkdir /root/bin; vim /root/bin/backup.sh
#!/bin/bash
backdir="/etc /home /root /var/spool/mail"
basedir=/backup
[ ! -d "$basedir" ] && mkdir $basedir
backfile=$basedir/backup.tar.gz
tar -zcvf $backfile $backdir

[root@localhost ~]# vim /etc/crontab
45 2 * * * root sh /root/bin/backup.sh
```

无论如何，以现今的网络功能及维护来看，架设一个『功能性强』的主机， 还不如架设一个『稳定且安全的主机』比较好一点！因此，对于主机的安全要求就需要严格的要求啦！就鸟哥的观点来看， 如果你的主机是用来替你赚钱的，例如某些研究单位的大型 Cluster 运算主机， 那么即使架设一个甚至让你觉得很不方便的防火墙系统，都是合理的手段！因为主机被入侵就算了，若数据被窃取，呵呵！ 那可不是闹着玩的！

由上面的整个架站流程来看，由规划到安装、主机设定、账号与档案权限管理、后续安全性维护与管理以及重要的备份工作等等， 必需要每个环节都很清楚，才能够设定出一个较为稳定而可正常工作的服务器。而上面的每一个工作都涉及到相当多的 Linux 基础操作与相关的概念，所以说，想要学架站，真的真的不能省略了 Linux 的基础学习， 这也是为什么我们一再强调 Linux 新手不要一头栽入想要单纯架设服务器的迷思当中呐！ 如果你对于上面谈到的几个基础概念不是很清楚的话，那么建议你由底下的两个网站学起：

- http://www.study-area.org
- http://linux.vbird.org

1.3 自我评估是否已经具有架站的能力

网管人员需要什么能力呢？我想，架几个站跟作一个称职的网管人员， 相差是甚远的！架站，说真的，是一件很简单的事情，看着书本一步一步的作上去，一定可以成功的！但是，很多人都只晓得 『如何架站』却不知到『如何维护一个网站的安全』！基本上， 维护一个已经架设好的网站的正常运作，真的要比架设一个网站难的多了！你得要随时知道你的系统状况， 随时注意是否有新的软件漏洞而去修补他，随时要注意各种服务的登录档案(logfile)以了解系统的运作情况！ 得知道发生问题的时候，到底问题点是在哪一个！

比如说当机了，那么你知道当机的原因吗？ 即使不知道，也可得需要约略猜得出来才行。而，如果安全出了问题，被入侵了，除了 format + 重灌之外，可有办法在不移除系统的情况下修补漏洞？ 这些都是网管人员需要学习的，而且，通常都是需要经验的累积才会知道问题的所在！ 此外，保持身心的活力以随时注意在线公布的安全防备信息等等！都需要具备的！

此外，最严重的问题是，网管人员其实最需要的是 『道德感与责任感』！你可要晓得你的机器上所有人的隐私都在你的监控之下， 如果你本身就已经有偷窥欲了，可知道这有多可怕吗？另外，如果没有责任感的人作为一个网管， 可能会疯掉，因为不论何时何地，只要是你监控的主机出了问题，嘿嘿嘿嘿，你一定是第一个被想到的人物， 所以，你得随时随地做好可能随时会被召唤回主机跟前的心理准备！

更可笑的是，如果你服务的人群中， 有几个连开机的时候软盘驱动器塞了一块不可开机的软盘，导致无法正常开机， 也都会跟你抱怨说『唉呦！你经手的计算机怎么这么烂，动不动就不能开机』的时候， 你得要有容人的雅量，说说冷笑话解解闷吧！总之，网管人员并不是只要会架站就可以了， 『道德感』『责任感』还有『耐心』呵呵！套一句现在人喜欢说的口头禅『这是一定要的啦！』

网管人员是什么？ 好久以前看到了报纸的一篇报导， 内容大概是说：台湾的网络管理人员对于『网络安全性防护』的认知不够，或许是防火墙机制建立不完整， 或者是认为黑客不会入侵小型网站，所以在不甚了解的情况下，被所谓的『中东黑客组织』所入侵， 然后以台湾被入侵的计算机为跳板，去攻击宾拉登的仇敌美国，然后引起美国高度的不满。由于台湾的立场有点得罪不得美国 (这边不提及政治因素，反正目前的情况是这样。) ，所以一接到美国来的抗议信函就很棘手。 这只是一个事件问题，不过这个事件问题也点出了一个重点，就是我们的网络信息可能真的是蛮发达的， 不过，管理网络的人员可能在认知的程度上就有点参差不齐了！网络安全是蛮重要的，只是， 大家常常会忘记他！个人认为，网管是蛮重要的角色，应该不能等闲视之才对。

好了，如果你了解了上面鸟哥所想要表达的意念之后，来评估看看你是否适合当一个称职的网管人员吧！

1. 是否具有 Linux 的基础概念： 这当然包含很多部分，例如账号管理、BASH、权限的概念、Process 与 signal 的概念、简易的硬件与 Linux 相关性 (如 mount)的认识、登录档案的解析、daemon 的认识等等，都需要有一定程度的了解；

2. 是否具备基础网络知识： 没有网络知识想要架站，那是天方夜谭！请确认你已经熟悉 IP, Netmask, route, DNS, daemon 与 port, TCP 封包的概念等基本知识；
3. 是否已经身心活化了： 网管人员必须要随时注意网站的相关信息，这包括网站软件的漏洞修补、 网络上公告的网络安全通报等等，还有，得要每日分析主机的登录文件， 你是否已经具备了随时注意这些信息的『耐心』呢？
4. 是否具有道德感与责任感： 如果还是具有一点点的偷窥欲，再加油吧！^_^ ，另外，如果老板想要请你『偷窥』时，请想尽任何方法，让他理解这么做是多么的可笑～

当然，一再强调的，架设一个 Linux 服务器是很简单的，但是维护的工作除了身心已经活化， 并且还要拥有高标准的道德感，否则.....倒站恐怕是可以预见的一个后果.....

1.4 本章习题

- 本章所安装的 samba 软件未来还会使用到，因此请先移除 samba 软件，并将本章例题中改写的 /etc/crontab 内容取消 (共两行)。透过 yum remove samba 或 rpm -e samba 均可，然后用 vim /etc/crontab 将那两行取消吧！
- 如果我有一颗硬盘在 A 主机上面安装了 Linux 之后，拿到另一台配备相同的 B 主机上面去进行开机，结果竟然无法顺利开机，你认为可能的原因是什么？不能开机常常是因为找不到根目录的位置，而根目录找不到通常就是磁盘的装置文件名错误所致。目前由于 /etc/fstab 配合 filesystem 都使用 LABEL name ，所以不容易发生这样的情况。但如果你曾经自行手动处理过 /etc/fstab 的话，那就必须要注意磁盘的装置文件名了！ 透过修改 /etc/fstab 以及 /boot/grub/menu.lst 或许能够得到方法解决。
- 一般来说，在 Linux 系统上，用户默认的家目录在那个目录下？另外，新增一个使用者时， 该用户默认的家目录内容来自那个目录下？在 /etc/default/useradd 这个档案里面会规范用户的默认家目录以及默认家目录的内容，一般来说，用户默认家目录在 /home ，至于家目录内的档案则复制来源在 /etc/skel 里面。
- 我以原始码的方式进行一个软件的安装，但是在分析系统的时候，分析程序一直告诉我找不到 cc 这个指令，请问这是什么问题？为何需要 cc ？又，我该如何解决这个问题，好让软件可以顺利的被安装在我的 Linux 上面？因为是原始码，所以还需要编译程序来将该原始码编译成为可以在你的 Linux 系统上面跑的 binary 档案，在 Linux 上头默认的编译程序就是 gcc 这个编译程序(compiler)。如果你在安装 Linux 的时候，使用 Linux Installer 默认的软件选择，那通常会没有安装 gcc 以及 make 等软件，此时，请使用 yum 去处理软件的安装吧！
- 我发现我的 Linux 系统怪怪的，似乎有什么不知名的程序在内存当中跑，我该如何将这个不知名的程序捉出来，并且将他移除？如果要捉出程序(process)的话，可以使用 ps -aux 或者是直接输入 top 来查询 process 的 ID (PID)，找到 PID 号码后，再以 kill -9 PID 来删除该程序即可。
- 我总是无法编辑某个档案，你认为应该是什么问题造成的？那又要怎么解决？无法编辑某个档案，可以先使用 file 这个指令来查询一下该档案的格式，例如想察看 /etc/shadow 的格式，可以下达：『file /etc/shadow』，如果是文本文件，却还是无法编辑，那么最可能发生的原因就是『权限』的问题了。可以使用 ls -l filename 察看档案权限，再以 chmod 或 chown 来修订该档案的权限。此外，该档案也可能含有隐藏属性，可以使用 lsattr filename 查阅，再以 chattr 来修订隐藏属性。
- 你认为一个称职的网管人员应该具备什么能力？能力需求相当高，如了(1)操作系统的基础知识(不论是 Linux/Unix/MAC/MS)；(2)网络基础的知识；(3)个别 Internet Services 的运作知识之外，还需要(4)身心保持在备战状态，以及(5)具有相当高程度的道德感、责任感与使命感。
- 我要关掉 cron 这个服务，应该怎么关掉他？如果正常的方法无法关闭这个服务，可以使用什么方法来关闭？因为 cron 是一个 stand alone 的服务，所以可以使用 /etc/rc.d/init.d/cron stop 来关闭；如果还是无法正常关闭，可以使用 ps -aux | grep cron

捉出该程序的 PID ，然后以 kill -9 PID 来关闭。

- 如果一开机就要执行某个程序，应该要将该程序写入那个档案里面？可以直接在 /etc/rc.d/rc[run-level].d 里面加入 S 开头的档案，不过，更简单的作法是直接将该程序写入 /etc/rc.d/rc.local ，不过，请注意该程序必须要具有可执行的权限，且 rc.local 也必须要是可执行喔！

2003/07/30：第一次完成日期！ 2003/08/19：加入了课后练习，如果你无法回答上面的问题.....不要怀疑，赶紧回去参考 Linux 基础篇！ 2003/09/06：加入课后练习的参考用解答 2006/02/07：将原本的旧文移到此处 2006/06/06：将 SATA 接口的硬盘代号再次做个修订！目前 SATA 的格式有分两种呢！ 2007/01/02：将一些排版重整，将一些日期方面的数据重整，将课后练习补上来 2010/05/07：将 CentOS 4.x 为底的旧文章移动到 此处 2010/07/22：重新设计服务器安装流程，并且在每个基础数据都加上练习！尤其是全新安装一部 server 以供使用！ 2011/07/14：将原本基于 CentOS 5.x 为底的旧文章移动到此处 2011/07/14：将安装与设定的数据通通改为 CentOS 6.x 的版本啰！更新真困扰～@_@

第二章、基础网络概念

最近更新日期：2011/07/15

你的服务器是放在网络网络上面来提供服务的，所以，如果没有网络或者是网络不通，那么你的服务器当然是英雄无用武之地啦！ 此外，服务器上面的网络服务都是用来达成某项因特网的通讯协议，以提供相对应的服务而已。所以啰， 你当然得要知道这个最基础的网络概念，否则，当服务器的服务出现问题时，你该如何解决啊？您说对吧！ 这部份最重要的是 TCP/IP 与 OSI 七层协议的相关概念了，这部份难的很～难的很～ 在这一章中，鸟哥以较为口语的方式来介绍这些基础网络架构，希望能带给朋友们快速了解网络是啥。 当然，想要更了解网络相关功能的话，文末的参考资料可以参考看看喔！ ^_^

2.1 网络是个什么玩意儿

全世界的人种有很多，人类使用的语言种类也多的很。那如果你想要跟外国人沟通时，除了比手划脚之外，你要如何跟对方讲话？ 大概只有两种方式啰，一种是强迫他学中文，一种则是我们学他的语言，这样才能沟通啊。在目前世界上的强势语言还是属于英语系国家， 所以啰，不管是啥人种，只要学好英文，那么大家都讲英文，彼此就能够沟通了。希望不久的未来，咱们的中文能够成为强势语言啊！

这个观念延伸到网络上面也是行的通的，全世界的操作系统多的很，不是只有 Windows/Linux 而已，还有苹果计算机自己的操作系统， Unix like 的操作系统也非常多！那么多的操作系统 (人种) 要如何进行网络沟通 (语言) 呢？那就得要制订共同遵守的标准才行了。这个标准是由国际组织规范的，你的系统里面只要提供可以加入该标准的程序代码， 那你就能够透过这个标准与其他系统进行沟通！所以啰，网络是跨平台的，并不是只有 Linux 才这么做！因此，这部份的资料你学完后，是可以应用在所以操作系统上面的！观念都相同啊！

另外，这一个章节旨在引导网络新鲜人快速进入网络的世界，所以鸟哥写的比较浅显一些些，基本上， 还有一堆网络硬件与通讯协议并没有被包含在这篇短文里头。如果你的求知欲已经高过本章节， 那么请自行到书局寻找适合你自己的书籍来阅读！当然，你也可以在因特网上面找到你所需要的数据。 在本章最后的参考数据可以瞧一瞧呐！

2.1.1 什么是网络

我们都知道，网络就是几部计算机主机或者是网络打印机之类的接口设备， 透过网络线或者是无线网络的技术，将这些主机与设备连接起来， 使得数据可以透过网络媒体(网络线以及其他网络卡等硬件)来传输的一种方式。 请你想象一下，如果你家里面只有计算机、打印机、传真机等机器，却没有网络连接这些硬件， 那么使用上会不会很麻烦？如果将这个场景移到需要工作的办公室时， 计算机的数据无法使用网络连接到打印机来打印，那是否很伤脑筋呢？对吧！光用想的就觉得很麻烦吧！不幸的是，这些麻烦事在 1970 年代以前，确实是存在的啊！

- 各自为政的『网络硬件与软件』技术发展： Ethernet & Token-Ring

  在 1970 年代前后，为了解决这个烦人的数据传输问题，各主要信息相关的公司都在研究各自的网络连接技术， 以使自家的产品可以在办公室的环境底下组织起来。其中比较有名的就是全录公司的 Ethernet 技术， 以及 IBM 研发的 Token-Ring 技术了。但是这些技术有个很大的问题，那就是它们彼此不认识对方的网络技术！ 也就是说，万一你的办公室购买了整合 Ethernet 技术的计算机主机，但是其他的计算机却是使用 IBM 的机器时，想要在这两者之间进行数据的沟通，在早期来说那是不可能的。

- 以『软件』技术将硬件整合： ARPANET & TCP/IP

  为了解决上述的网络硬件整合功能，所以在 1960 年代末期美国国防部就开始研究一个可以在这些不同的网络硬件上面运作的软件技术， 使得不同公司的计算机或数据可以透过这个软件来达成数据沟通。这个研究由美国国防部尖端研究企画署 (Defense Advanced Research Project Agency, DARPA) 负责，他们将该网络系统称为 ARPANET， 这个咚咚就是目前熟知的 TCP/IP 技术的雏形了！在 1975 年左右， ARPANET 已可以在常见的 Ethernet 与 Token-Ring 等硬件平台底下互通数据了。DARPA 在 1980 年正式推出 TCP/IP 技术后， 由于想要推展此项技术，因此与柏克莱 (Berkeley) 大学合作，将 TCP/IP 植入著名的 BSD Unix 系统内， 由于大学乃是未来人才数据库的培养处，所以，TCP/IP 这项技术便吸引越来越多使用者的投入， 而这种连接网络的技术也被称之为 Internet (注1)。

- 没有任何王法的因特网： Internet

  现在我们知道 Internet 就是使用 TCP/IP 的网络连接技术所串联起来的一个网络世界，而这个 Internet 在 1980 年代之后由于对 email 的需求以及浏览器图形接口的兴起，因此快速的蔓延在计算机世界中。 但是， Internet 有没有人在管理啊？很不巧的是， Internet 是一个管理相当松散的所在。 只要你能够使用任何支持 TCP/IP 技术的硬件与操作系统，并且实际连接上网络后， 你就进入 Internet 的世界了。在该世界当中，没有任何王法的保护，你的实际数据如果接上 Internet ， 在任何时刻都需要自己保护自己，免得中了『流弹』而受伤啊！

  为甚么说 Internet 没有王法呢？这是因为 Internet 仅是提供一个网络的连接接口， 所以你只要连接上 Internet 后，全世界都可以任你遨游，不过也因为如此，『跨海』而来的攻击就成了简单的事件， 简单说，台湾的法律仅适用台湾地区对吧？但是计算机怪客 (cracker) 可以在国外透过 Internet 对你的主机进行攻击， 我们的法律可管不到国外地区啊！虽然可以透过很多国际管道来寻求协助，不过，还是很难协助你缉拿凶手的啊。 因此啰，在你的主机要连上 Internet 之前，请先询问自己，真的有需要连上 Internet 吗？ ^_^

- 软硬件标准制定的成功带来的影响： IEEE 标准规范

  现在我们常常听到『你要上网啊！那你要去买网络卡喔！还得要连接到 Internet 才行啊！』 这个网络卡就是市面上随处可见的一个适配卡而已，至于 Internet 则是去向 Hinet/Seed net 或 其他网络服务提供公司 (Internet Service Provider, ISP) 申请的账号密码。 问题是，是否就只有透过网络卡与 Internet 才能上网啊？呵呵！当然不是！其他不同的网络硬件与软件可多着那！ 不过，最成功的却是以太网络 (Ethernet) 与 Internet ，这是为甚么呢？这两者的技术比较好吗？ 当然不是！这是因为这两者都被『标准』所支持的缘故(注2)。

  以太网络最初是由全录公司 (Xerox PARC) 所建构出来的，而后透过 DEC, Intel 与 Xerox 合作将以太网络标准化。 再经由 IEEE (Institute of Electrical and Electronic Engineers 注3) 这个国际著名的专业组织利用一个 802 的项目制定出标准，之后有 19 家公司宣布

支持 IEEE 所发布的 802.3 标准， 并且到了 1989 年国际标准化组织 ISO (International Organization for Standard) 将以太网络编入 IS88023 标准， 呵呵！这表示以太网络已经是一项公认的标准接口了，如此一来，大家都可以依据这个标准来设定与开发自己的硬件， 只要硬件符合这个标准，理论上，他就能够加入以太网络的世界，所以，购买以太网络时， 仅需要查看这个以太网络卡支持哪些标准就能够知道这个硬件的功能有哪些，而不必知道这个以太网络卡是由哪家公司所制造的呐。

Tips: 标准真的是个很重要的东西，真要感谢这些维护标准的专业组织。当有公司想要开发新的硬件时， 它可以参考标准组织所发布与维护的文件资料，透过这些文件数据后，该公司就知道要制作的硬件需要符合哪些标准， 同时也知道如何设计这些硬件，让它可以『兼容』于目前的机器，让使用者不会无所适从啊。 包括软件也有标准，早期 Linux 在开发时就是透过了解 POSIX 这个标准来设计核心的， 也使得 Linux 上面可以执行大多数的标准接口软件呢！你说，标准是否真的很重要啊！




```
除了硬件之外，TCP/IP 这个 Internet 的通讯协议也是有标准的，这些标准大部分都以 RFC (Request For Comm
```

2.1.2 计算机网络组成组件

接下来，让我们来谈谈那么组成计算机网络的组件有哪些呢？这些组件的定义为何啊？我们得要先知道有哪些硬件嘛！ 接下来才好理解啊。在这里，我们以底下这张联机示意图来解释好了：

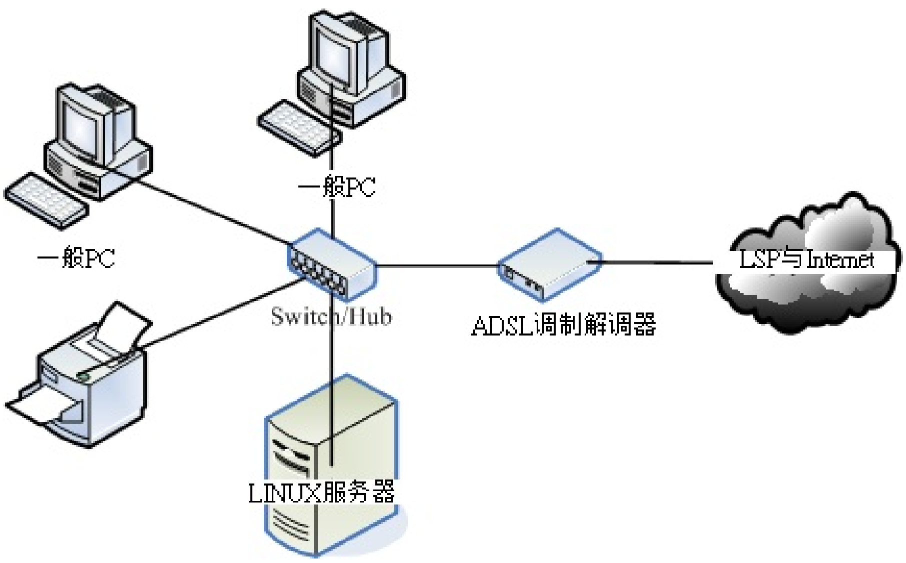


图 2.1-1、计算机网络联机示意图

在上图中，我们主要需要注意到的硬件有哪些呢？大致有底下这些啦：

- 节点 (node)：节点主要是具有网络地址 (IP) 的设备之称， 因此上面图示中的一般PC、Linux服务器、ADSL调制解调器与网络打印机等，个别都可以称为一个 node ！ 那中间那个集线器 (hub) 是不是节点呢？因为他不具有 IP ，因此 hub 不是节点。
- 服务器主机 (server)：就网络联机的方向来说，提供数据以『响应』给用户的主机， 都可以被称为是一部服务器。举例来说，Yahoo 是个 WWW 服务器，昆山的 FTP (http://ftp.ksu.edu.tw/) 是个文件服务器等等。
- 工作站 (workstation) 或客户端 (client)：任何可以在计算机网络输入的设备都可以是工作站， 若以联机发起的方向来说，主动发起联机去『要求』数据的，就可以称为是客户端 (client)。举例来说，一般 PC 打开浏览器对 Yahoo 要求新闻数据，那一般 PC 就是客户端。
- 网络卡 (Network Interface Card, NIC)：内建或者是外插在主机上面的一个设备， 主要提供网络联机的卡片，目前大都使用具有 RJ-45 接头的以太网络卡。一般 node 上都具有一个以上的网络卡， 以达成网络联机的功能。
- 网络接口：利用软件设计出来的网络接口，主要在提供网络地址 (IP) 的任务。 一张网卡至少可以搭配一个以上的网络接口；而每部主机内部其实也都拥有一个内部的网络接口，那就是 loopback (lo) 这个循环测试接口！
- 网络形态或拓朴 (topology)：各个节点在网络上面的链接方式，一般讲的是物理连接方式。 举例来说，上图中显示的是一种被称为星形联机 (star) 的方式，主要是透过一个中间连接设备， 以放射状的方式连接各个节点的一种形态，这就是一种拓朴。
- 网关 (route) 或通讯闸 (gateway)：具有两个以上的网络接口， 可以连接两个以上不同的网段的设备，例如 IP 分享器就是一个常见的网关设备。那上面的 ADSL 调制解调器算不算网关呢？ 其实不太能算，因为调制解调器通常视为一个在主机内的网卡设备，我们可以在一般 PC 上面透过拨号软件， 将调制解调器仿真成为一张实体网卡 (ppp) ，因此他不太能算是网关设备啦！

网络设备其实非常多也非常复杂，不过如果以小型企业角度来看，我们能够了解上述图示内各设备的角色，那应该也足够啰！ 接下来，让我们继续来讨论一下网络范围的大小吧！

2.1.3 计算机网络区域范围

由于各个节点的距离不同，联机的线材与方式也有所差异，由于线材的差异也导致网络速度的不同，让网络的应用方向也不一样。 根据这些差异，早期我们习惯将网络的大小范围定义如下：(注6)

- 局域网络 (Local Area Network, LAN)： 节点之间的传输距离较近，例如一栋大楼内，或一个学校的校区内。可以使用较为昂贵的联机材料， 例如光纤或是高质量网络线 (CAT 6) 等。网络速度较快，联机质量较佳且可靠，因此可应用于科学运算的丛集式系统、 分布式系统、云端负荷分担系统等。

- 广域网 (Wide Area Network, WAN)： 传输距离较远，例如城市与城市之间的距离，因此使用的联机媒体需要较为便宜的设备，例如经常使用的电话线就是一例。 由于线材质量较差，因此网络速度较慢且可靠性较低一些，网络应用方面大多为类似 email, FTP, WWW 浏览等功能。

除了这两个之外，还有所谓的都会网络 (Metropolitan Area Network, MAN)，不过近来比较少提及，因此你只要知道有 LAN 及 WAN 即可。这两个名词在很多地方你都可以看的到喔！改天你回家看看你家的 ADSL 调制解调器或 IP 分享器后面的插孔看看，你就能够看到有 WAN 与 LAN 的插孔，现在你就知道为啥有这两个灯号与插孔了吧。

一般来说，LAN 指的是区域范围较小的环境，例如一栋大楼或一间学校，所以在我们生活周遭有着许许多多的 LAN 存在。 那这些 LAN 彼此串接在一起，全部的 LAN 串在一块就是一个大型的 WAN 啰！简单的说，就是这样分。

不过，现在的环境跟以前不一样了，举例来说，前几天刚刚宣布 (2011/07)，光纤的速度已经可以到达 100Mbps/10Mbps 的下载/上传带宽了！再举例来说，台湾的学术网络通通是串在一块的，鸟哥在台南昆山联机到高雄义守大学下载 CentOS 映像档时，你猜下载的速度有多快？每秒钟可高达 100Mbps 左右！这已经是一个内部区网的速度了！所以，用以前的观点来看， 其实对目前的网络环境有点不符现象了。因此，目前你可以使用『速度』作为一个网络区域范围的评量。 或许现在我们可以说，整个台湾的学术网络 (TANET, 注7) 可以视为是一个局域网络呢！

2.1.4 计算机网络协议： OSI 七层协定

谈完了网络需要制订的标准、网络联机的组件以及网络的范围之后，接下来就是要讲到，那么各个节点之间是如何沟通讯息的呢？ 其实就是透过标准的通讯协议啦！但是，整个网络连接的过程相当复杂，包括硬件、软件数据封包与应用程序的互相链接等等， 如果想要写一支将联网全部功能都串连在一块的程序，那么当某个小环节出现问题时，整只程序都需要改写啊！真麻烦！

那怎办？没关系，我们可以将整个网络连接过程分成数个阶层 (layer)，每个阶层都有特别的独立的功能， 而且每个阶层的程序代码可以独立撰写，因为每个阶层之间的功能并不会互相干扰的。 如此一来，当某个小环节出现问题时，只要将该层级的程序代码重新撰写即可。所以程序撰写也容易，整个网络概念也就更清晰！ 那就是目前你常听到的 OSI 七层协定 (Open System Interconnection) 的概念啰！

如果以图示来说，那么这七个阶层的相关性有点像底下这样：

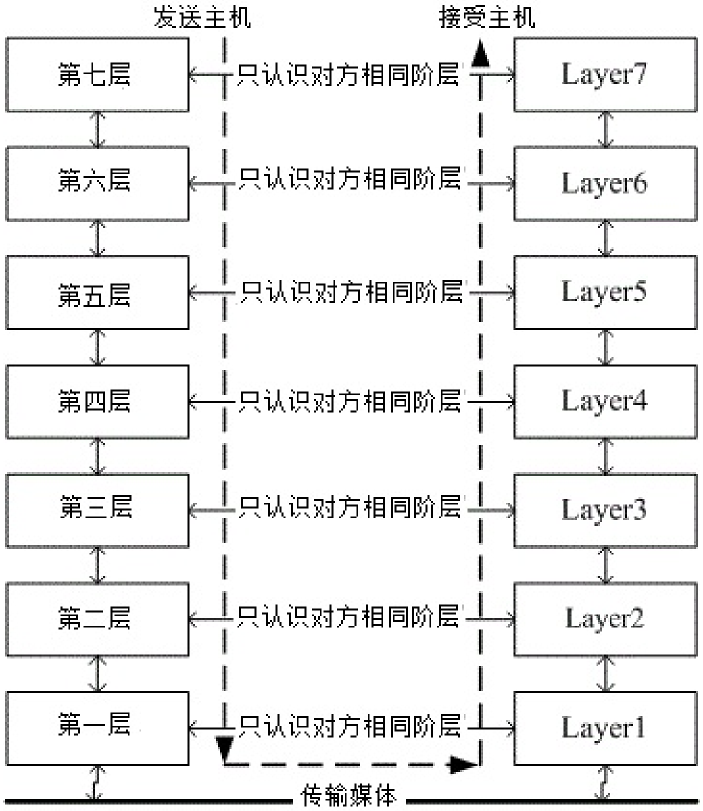


图 2.1-2、OSI 七层协议各阶层的相关性

依据定义来说，越接近硬件的阶层为底层 (layer 1)，越接近应用程序的则是高层 (layer 7)。不论是接收端还是发送端，每个一阶层只认识对方的同一阶层数据。 而整个传送的过程就好像人们在玩整人游戏一般，我们透过应用程序将数据放入第七层的包裹，再将第七层的包裹放到第六层的包裹内， 依序一直放到第一层的最大的包裹内，然后传送出去给接收端。接收端的主机就得由第一个包裹开始，依序将每个包裹拆开， 然后一个一个交给对应负责的阶层来视察！这就是整人游戏...喔！是 OSI 七层协议在阶层定义方面需要注意的特色。

既然说是包裹，那我们都知道，包裹表面都会有个重要的信息，这些信息包括有来自哪里、要去哪里、接收者是谁等等， 而包裹里面才是眞正的数据。同样的，在七层协议中，每层都会有自己独特的表头数据 (header)，告知对方这里面的信息是什么， 而眞正的数据就附在后头啰！我们可以使用如下的图示来表示这七层每一层的名字，以及数据是如何放置到每一层的包裹内：

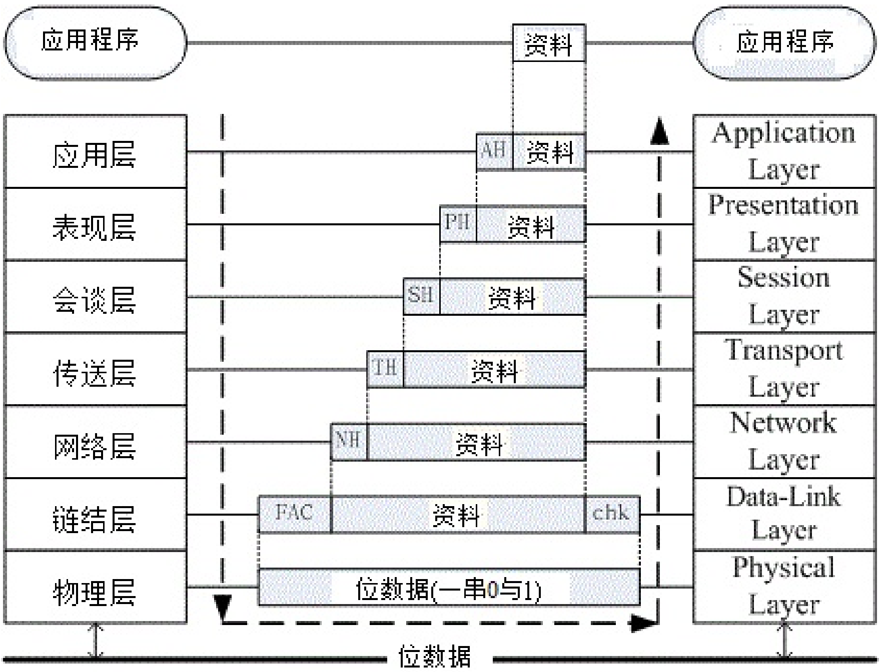


图 2.1-3、OSI 七层协议数据的传递方式

上图中仔细看每个数据报的部分，上层的包裹是放入下层的数据中，而数据前面则是这个数据的表头。其中比较特殊的是第二层， 因为第二层 (数据链结层) 主要是位于软件封包 (packet) 以及硬件讯框 (frame) 中间的一个阶层， 他必须要将软件包装的包裹放入到硬件能够处理的包裹中，因此这个阶层又分为两个子层在处理相对应的数据。 因为比较特殊，所以您瞧瞧，第二层的数据格式比较不一样喔，尾端还出现一个检查码哩～

每一个阶层所负责的任务是什么呢？简单的说，每一层负责的任务如下：(注6, 注8, 注9)

分层	负责内容
Layer 1 物理层 Physical Layer	由于网络媒体只能传送 0 与 1 这种位串，因此物理层必须定义所使用的媒体设备之电压与讯号等， 同时还必须了解数据讯框转成位串的编码方式，最后连接实体媒体并传送/接收位串。
Layer 2 数据链结层 Data-Link Layer	这一层是比较特殊的一个阶层，因为底下是实体的定义，而上层则是软件封装的定义。因此第二层又分两个子层在进行数据的转换动作。 在偏硬件媒体部分，主要负责的是 MAC (Media Access Control) ，我们称这个数据报裹为 MAC 讯框 (frame)， MAC 是网络媒体所能处理的主要数据报裹，这也是最终被物理层编码成位串的数据。MAC 必须要经由通讯协议来取得媒体的使用权， 目前最常使用的则是 IEEE 802.3 的以太网络协议。详细的 MAC 与以太网络请参考下节说明。至于偏向软件的部分则是由逻辑链接层 (logical link control, LLC) 所控制，主要在多任务处理来自上层的封包数据 (packet) 并转成 MAC 的格式， 负责的工作包括讯息交换、流量控制、失误问题的处理等等。
Layer 3 网络层 Network Layer	这一层是我们最感兴趣的啰，因为我们提及的 IP (Internet Protocol) 就是在这一层定义的。 同时也定义出计算机之间的联机建立、终止与维持等，数据封包的传输路径选择等等，因此这个层级当中最重要的除了 IP 之外，就是封包能否到达目的地的路由 (route) 概念了！
Layer 4 传送层 Transport Layer	这一个分层定义了发送端与接收端的联机技术(如 TCP, UDP 技术)， 同时包括该技术的封包格式，数据封包的传送、流程的控制、传输过程的侦测检查与复原重新传送等等， 以确保各个数据封包可以正确无误的到达目的端。
Layer 5 会谈层 Session Layer	在这个层级当中主要定义了两个地址之间的联机信道之连接与挂断，此外，亦可建立应用程序之对谈、 提供其他加强型服务如网络管理、签到签退、对谈之控制等等。如果说传送层是在判断资料封包是否可以正确的到达目标， 那么会谈层则是在确定网络服务建立联机的确认。
Layer 6 表现层 Presentation Layer	我们在应用程序上面所制作出来的数据格式不一定符合网络传输的标准编码格式的！ 所以，在这个层级当中，主要的动作就是：将来自本地端应用程序的数据格式转换(或者是重新编码)成为网络的标准格式， 然后再交给底下传送层等的协议来进行处理。所以，在这个层级上面主要定义的是网络服务(或程序)之间的数据格式的转换， 包括数据的加解密也是在这个分层上面处理。
Layer 7 应用层 Application Layer	应用层本身并不属于应用程序所有，而是在定义应用程序如何进入此层的沟通接口，以将数据接收或传送给应用程序，最终展示给用户。

事实上， OSI 七层协议只是一个参考的模型 (model)，目前的网络社会并没有什么很知名的操作系统在使用 OSI 七层协定的联网程序代码。那...讲这么多干嘛？这是因为 OSI 所定义出来的七层协议在解释网络传输的情况来说， 可以解释的非常棒，因此大家都拿 OSI 七层协议来做为网络的教学与概念的理解。至于实际的联网程序代码，那就交给 TCP/IP 这个玩意儿吧！

2.1.5 计算机网络协议： TCP/IP

虽然 OSI 七层协议的架构非常严谨，是学习网络的好材料。但是也就是因为太过严谨了，因此程序撰写相当不容易， 所以造成它在发展上面些许的困扰。而由 ARPANET 发展而来的 TCP/IP 又如何呢？其实 TCP/IP 也是使用 OSI 七层协议的观念， 所以同样具有分层的架构，只是将它简化为四层，在结构上面比较没有这么严谨，程序撰写会比较容易些。后来在 1990 年代由于 email, WWW 的流行，造成 TCP/IP 这个标准为大家所接受，这也造就目前我们的网络社会啰！

既然 TCP/IP 是由 OSI 七层协议简化而来，那么这两者之间有没有什么相关性呢？它们的相关性可以图示如下， 同时这里也列出目前在这架构底下常见的通讯协议、封包格式与相关标准：

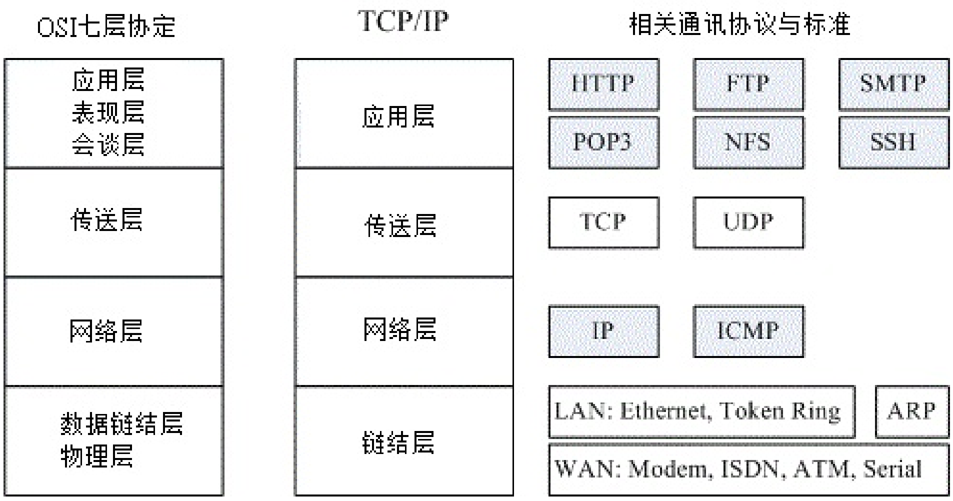


图 2.1-4、OSI 与 TCP/IP 协议之相关性

从上图中，我们可以发现 TCP/IP 将应用、表现、会谈三层整合成一个应用层，在应用层上面可以实作的程序协议有 HTTP, SMTP, DNS 等等。 传送层则没有变，不过依据传送的可靠性又将封包格式分为连接导向的 TCP 及非连接导向的 UDP 封包格式。网络层也没有变，主要内容是提供了 IP 封包，并可选择最佳路由来到达目标 IP 地址。数据链结层与物理层则整合成为一个链结层，包括定义硬件讯号、 讯框转位串的编码等等，因此主要与硬件 (不论是区网还是广域网) 有关。

那 TCP/IP 是如何运作的呢？我们就拿妳常常连上的 Yahoo 入口网站来做个说明好了，整个联机的状态可以这样看：

1. 应用程序阶段：妳打开浏览器，在浏览器上面输入网址列，按下 [Enter]。此时网址列与相关数据会被浏览器包成一个数据， 并向下传给 TCP/IP 的应用层；
2. 应用层：由应用层提供的 HTTP 通讯协议，将来自浏览器的数据报起来，并给予一个应用层表头，再向传送层丢去；
3. 传送层：由于 HTTP 为可靠联机，因此将该数据丢入 TCP 封包内，并给予一个 TCP 封包的表头，向网络层丢去；
4. 网络层：将 TCP 包裹包进 IP 封包内，再给予一个 IP 表头 (主要就是来源与目标的 IP 啰)，向链结层丢去；
5. 链结层：如果使用以太网络时，此时 IP 会依据 CSMA/CD 的标准，包裹到 MAC 讯框

中，并给予 MAC 表头，再转成位串后， 利用传输媒体传送到远程主机上。

等到 Yahoo 收到你的包裹后，在依据相反方向拆解开来，然后交给对应的层级进行分析，最后就让 Yahoo 的 WWW 服务器软件得到你所想要的数据，该服务器软件再根据你的要求，取得正确的资料后，又依循上述的流程，一层一层的包装起来， 最后传送到你的手上！就是这样啰！

根据这样的流程，我们就得要知道每个分层所需要了解的基础知识，这样才算学习网络基础嘛！所以底下我们会依据 TCP/IP 的链结层、网络层、传送层来进行说明，应用层的协议则在后续章节中有对应的协定再来谈啰！同时我们也知道， 网络媒体一次传输的数据量是有限的，因此如果要被传输的数据太大时，我们在分层的包装中，就得要将数据先拆开放到不同的包裹中， 再给包裹一个序号，好让目的端的主机能够藉由这些序号再重新将数据整合回来！很有趣吧！接下来就让我们一层一层来介绍啰！

Tips: 一般来说，因为应用程序与程序设计师比较有关系，而网络层以下的数据则主要是操作系统提供的，因此， 我们又将 TCP/IP 当中的应用层视为使用者层，而底下的三层才是我们主要谈及的网络基础！所以这个章节主要就是介绍这三层啦！



2.2 TCP/IP 的链结层相关协议

TCP/IP 最底层的链结层主要与硬件比较有关系，因此底下我们主要介绍一些 WAN 与 LAN 的硬件。 同时会开始介绍那重要的 CSMA/CD 的以太网络协议，以及相关的硬件与 MAC 讯框格式等。那就开始来聊聊啰！

2.2.1 广域网使用的设备

在 2.1.3 节我们有提到过，广域网使用的设备价格较为低廉。 不过广域网使用到的设备非常的多，一般用户通常会接触到的主要是 ADSL 调制解调器或者是光纤到大厦，以及第四台的 Cable 宽带等。在这里我们先介绍一些比较常见的设备，如果以后你有机会接触到其他设备，再请你依据需求自行查阅相关书籍吧！

- 传统电话拨接：透过 ppp 协议

  早期网络大概都只能透过调制解调器加上电话线以及计算机的九针串行端口 (以前接鼠标或游戏杆的插孔)，然后透过 Point-to-Point Protocol (PPP 协议) 配合拨接程序来取得网络 IP 参数，这样就能够上网了。不过这样的速度非常慢，而且当电话拨接后， 就不能够讲电话了！因为 PPP 支持 TCP/IP, NetBEUI, IPX/SPX 等通讯协议，所以使用度非常广！

- 整合服务数字网络 (Integrated Services Digital Network, ISDN)

  也是利用现有的电话线路来达成网络联机的目的，只是联机的两端都需要有 ISDN 的调制解调器来提供联机功能。 ISDN 的传输有多种通道可供使用，并且可以将多个信道整合应用，因此速度可以成倍成长。基本的 B 信道速度约为 64Kbps， 但如美国规格使用 23 个以上的通道来达成联机，此时速度可达 1.5Mbps 左右。不过台湾这玩意儿比较少见。

- 非对称数位用路回路 (Asymmetric Digital Subscriber Line, ADSL)：透过 pppoe 协定

  也是透过电话线来拨接后取得 IP 的一个方法，只不过这个方式使用的是电话的高频部分，与一般讲电话的频率不同。 因此妳可以一边使用 ADSL 上网同时透过同一个电话号码来打电话聊天。在台湾，由于上传/下载的带宽不同， 因此才称为非对称的回路。ADSL 同样使用调制解调器，只是他透过的是 PPPoE (PPP over Ethernet) 的方法！ 将 PPP 仿真在以太网络卡上，因此你的主机需要透过一张网络卡来连接到调制解调器，并透过拨接程序来取得新的接口 (ppp0) 喔！

- 电缆调制解调器 (Cable modem)

主要透过有线电视 (台湾所谓的第四台) 使用的缆线作为网络讯号媒体，同样需要具备调制解调器来连接到 ISP，以取得网络参数来上网。 Cable modem 的带宽主要是分享型的，所以通常具有区域性，并不是你想装就能装的哩！

2.2.2 局域网络使用的设备-以太网络

在局域网络的环境中，我们最常使用的就是以太网络。当然啦，在某些超高速网络应用的环境中， 还可能会用到价格相当昂贵的光纤信道哩。只是如同前面提到的，以太网络因为已经标准化了，设备设置费用相对低廉， 所以一般你会听到什么网络线或者是网络媒体，几乎都是使用以太网络来架设的环境啦！ 只是这里还是要提醒您，整个网络世界并非仅有以太网络这个硬件接口喔！ 事实上，想了解整个以太网络的发展，建议你可以直接参考风信子与张民人先生翻译的 『Switched & Fast 以太网络』一书，该书内容相当的有趣，挺适合阅读的呐。底下我们仅做个简单的介绍而已。

- 以太网络的速度与标准

以太网络的流行主要是它成为国际公认的标准所致。早先 IEEE 所制订的以太网络标准为 802.3 的 IEEE 10BASE5 ，这个标准主要的定义是：『10 代表传输速度为 10Mbps，BASE 表示采用基频信号来进行传输，至于 5 则是指每个网络节点之间最长可达 500 公尺。』

由于网络的传输信息就是 0 与 1 啊，因此，数据传输的单位为每秒多少 bit ， 亦即是 M bits/second, Mbps 的意思。那么为何制订成为 10Mbps 呢？ 这是因为早期的网络线压制的方法以及相关的制作方法，还有以太网络卡制作的技术并不是很好， 加上当时的数据传输需求并没有像现在这么高，所以 10Mbps 已经可以符合大多数人的需求了。

Tips: 我们看到的网络提供者 (Internet Services Provider, ISP) 所宣称他们的 ADSL 传输速度可以达到 下行/上行 2Mbps/128Kbps (Kbits per second) 时，那个 Kb 指的可不是 bytes 而是 bits 喔！所以 2M/128K 在实际的档案大小传输速度上面，最大理论的传输为 256KBps/16 KBps(KBytes per second)，所以正常下载的速度约在每秒 100~200 KBytes 之间呐！ 同样的道理，在网络卡或者是一些网络媒体的广告上面，他们都会宣称自己的产品可以自动辨识传输速度为 10/100 Mbps (Mega-bits per second)，呵呵！该数值还是得再除以 8 才是我们一般常用的档案容量计算的单位 bytes 喔！



早期的网络线使用的是旧式的同轴电缆线，这种线路在现在几乎已经看不到了。取而代之的是类似传统电话线的双绞线 (Twisted Pair Ethernet) ，IEEE 并将这种线路的以太网络传输方法制订成为 10BASE-T 的标准。 10BASE-T 使用的是 10 Mbps 全速运作且采用无遮蔽式双绞

线 (UTP) 的网络线。此外，10BASE-T 的 UTP 网络线可以使用星形联机(star)，也就是以一个集线器为中心来串连各网络设备的一个方法，图 2.1-1 就是星形联机的一个示意图。

不同于早期以一条同轴电缆线链接所有的计算机的 bus 联机，透过星形联机的帮助，我们可以很简单的加装其他的设备或者是移除其他设备，而不会受到其他装置的影响，这对网络设备的扩充性与除错来说，都是一项相当棒的设计！也因此 10BASE-T 让以太网络设备的销售额大幅提升啊！

后来 IEEE 更制订了 802.3u 这个支持到 100Mbps 传输速度的 100BASE-T 标准，这个标准与 10BASE-T 差异不大，只是双绞线线材制作需要更精良，同时也已经支持使用了四对绞线的网络线了，也就是目前很常见的八蕊网络线呐！这种网络线我们常称为等级五 (Category 5, CAT5) 的网络线。这种传输速度的以太网络就被称为 Fast ethernet 。至于目前我们常常听到的 Gigabit 网络速度 1000 Mbps 又是什么呐？那就是 Gigabit ethernet 哩！只是 Gigabit ethernet 的网络线就需要更加的精良。

名称	速度	网络线等级
以太网络(Ethernet)	10Mbps	-
高速以太网络(Fast Ethernet)	100Mbps	CAT 5
超高速以太网络(Gigabit Ethernet)	1000Mbps	CAT 5e/CAT 6

为什么每当传输速度增加时，网络线的要求就更严格呢？这是因为当传输速度增加时，线材的电磁效应相互干扰会增强，因此在网络线的制作时就得需要特别注意线材的质料以及内部线蕊心之间的缠绕情况配置等，以使电子流之间的电磁干扰降到最小，才能使传输速度提升到应有的 Gigabit 。所以说，在以太网络世界当中，如果你想要提升原有的 fast ethernet 到 gigabit ethernet 的话，除了网络卡需要升级之外，主机与主机之间的网络线，以及连接主机线路的集线器/交换器等，都必须要提升到可以支持 gigabit 速度等级的设备才行喔！

- 以太网络的网络线接头 (跳线/并行线)

前面提到，网络的速度与线材是有一定程度的相关性的，那么线材的接头又是怎样呢？目前在以太网络上最常见到的接头就是 RJ-45 的网络接头，共有八蕊的接头，有点像是胖了的电话线接头，如下所示：

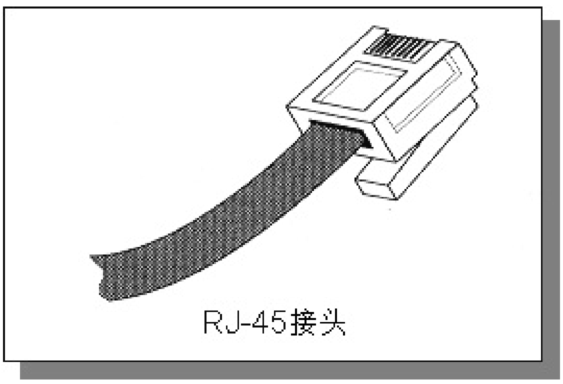


图 2.2-1、RJ-45 接头示意图

而 RJ-45 接头又因为每条蕊线的对应不同而分为 568A 与 568B 接头，这两款接头内的蕊线对应如下表：

接头名称\蕊线顺序	1	2	3	4	5	6	7	8
568A	白绿	绿	白橙	蓝	白蓝	橙	白棕	棕
568B	白橙	橙	白绿	蓝	白蓝	绿	白棕	棕

事实上，虽然目前的以太网络线有八蕊且两两成对，但实际使用的只有 1,2,3,6 蕊而已， 其他的则是某些特殊用途的场合才会使用到。但由于主机与主机的联机以及主机与集线器的联机时， 所使用的网络线脚位定义并不相同，因此由于接头的不同网络线又可分为两种：

- 跳线：一边为 568A 一边为 568B 的接头时称为跳线，用在直接链接两部主机的网络卡。
- 并行线：两边接头同为 568A 或同为 568B 时称为并行线，用在链接主机网络卡与集线器之间的线材；

2.2.3 以太网络的传输协议：CSMA/CD

整个以太网络的重心就是以太网络卡啦！所以说，以太网络的传输主要就是网络卡对网络卡之间的数据传递而已。 每张以太网络卡出厂时，就会赋予一个独一无二的卡号，那就是所谓的 MAC (Media Access Control) 啦！ 理论上，网卡卡号是不能修改的，不过某些笔记本电脑的网卡卡号是能够修改的呦！ 那么以太网络的网卡之间数据是如何传输的呢？那就得要谈一下 IEEE 802.3 的标准 CSMA/CD (Carrier Sense Multiple Access with Collision Detection) 了！我们以下图来作为简介，下图内的中心点为集线器， 各个主机都是联机到集线器，然后透过集线器的功能向所有主机发起联机的。

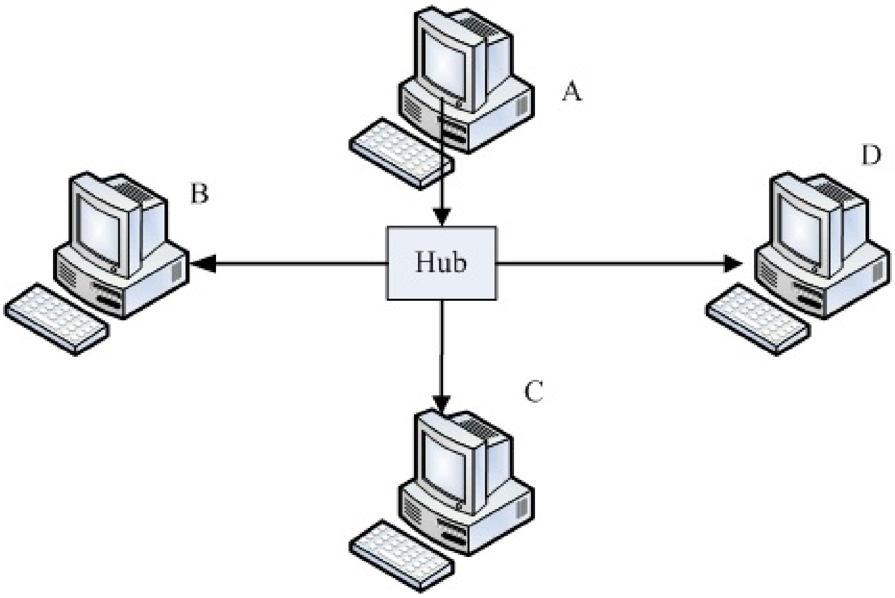


图 2.2-2、CSMA/CD联机示意图，由 A 发送资料给 D 时，注意箭头方向

集线器是一种网络共享媒体，什么是网络共享媒体啊？想象一下上述的环境就像一个十字路口，而集线器就是那个路口！ 这个路口一次只允许一辆车通过，如果两辆车同时使用这个路口，那么就会发生碰撞的车祸事件啊！那就是所谓的共享媒体。 也就是说，网络共享媒体在单一时间点内， 仅能被一部主机所使用。

理解了共享媒体的意义后，再来，我们就得要讨论，那么以太网络的网卡之间是如何传输的呢？我们以上图中的 A 要发给 D 网卡为例好了，简单的说， CSMA/CD 搭配上述的环境，它的传输情况需要有以下的流程：

1. 监听媒体使用情况 (Carrier Sense)：A 主机要发送网络封包前，需要先对网络媒体进行监听，确认没有人在使用后， 才能够发送出讯框；
2. 多点传输 (Multiple Access)：A 主机所送出的数据会被集线器复制一份，然后传送给所有连接到此集线器的主机！ 也就是说， A 所送出的数据， B, C, D 三部计算机都能够接收的到！但由于目标是 D 主机，因此 B 与 C 会将此讯框数据丢弃，而 D 则会抓下来处理；
3. 碰撞侦测 (Collision Detection)：该讯框数据附有检测能力，若其他主机例如 B 计算机也刚好在同时间发送讯框数据时， 那么 A 与 B 送出的数据碰撞在一块 (出车祸) ，此时这些讯框就是损毁，那么 A 与 B 就会各自随机等待一个时间， 然后重新透过第一步再传送一次该讯框数据。

了解这个程序很重要吗？我们就来谈谈：

- 网络忙碌时，集线器灯号闪个不停，但我的主机明明没有使用网络： 透过上述的流程我们会知道，不管哪一部主机发送出讯框，所有的计算机都会接收到！因为集线器会复制一份该数据给所有计算机。 因此，虽然只有一部主机在对外联机，但是在集线器上面的所有计算机灯号就都会闪个不停！
- 我的计算机明明没有被入侵，为何我的数据会被隔壁的计算机窃取： 透过上述的流程，我们只要在 B 计算机上面安装一套监听软件，这套软件将原本要丢弃的讯框数据捉下来分析，并且加以重组， 就能够知道原本 A 所送出的讯息了。这也是为什么我们都建议重要数据在因特网上面得要『加密』后再传输！
- 既然共享媒体只有一个主机可以使用，为何大家可以同时上网： 这个问题就有趣了，既然共享媒体一次只能被一个主机所使用，那么万一我传输 100MB 的档案，集线器就得被我使用 80 秒 (以 10Mbps 传输时)，在这期间其他人都不可以使用吗？不是的，由于标准的讯框数据在网络卡与其他以太网络媒体一次只能传输 1500bytes，因此我的 100MB 档案就得要拆成多个小数据报，然后一个一个的传送，每个数据报传送前都要经过 CSMA/CD 的机制。 所以，这个集线器的使用权是大家抢着用的！即使只有一部主机在使用网络媒体时，那么这部主机在发送每个封包间， 也都是需要等待一段时间的 (96 bit time)！

- 讯框要多大比较好？能不能修改讯框？： 如上所述，那么讯框的大小能不能改变呢？因为如果讯框的容量能够增大，那么小数据报的数量就会减少， 那每个讯框传送间的等待就可以减少了！是这样没错，但是以太网络标准讯框确实定义在 1500 bytes， 但近来的超高速以太网络媒体有支持 Jumbo frame (巨型讯框,注10) 的话，那么就能够将讯框大小改为 9000bytes 哩！但不是很建议大家随便修改啦！为什么呢？2.2.5 MTU 那小节再说。

2.2.4 MAC 的封装格式

上面提到的 CSMA/CD 传送出去的讯框数据，其实就是 MAC 啦！MAC 其实就是我们上面一直讲到的讯框 (frame) 啰！ 只是这个讯框上面有两个很重要的数据，就是目标与来源的网卡卡号，因此我们又简称网卡卡号为 MAC 而已。 简单的说，你可以把 MAC 想成是一个在网络线上面传递的包裹，而这个包裹是整个网络硬件上面传送数据的最小单位了。 也就是说，网络线可想成是一条『一次仅可通过一个人』的独木桥， 而 MAC 就是在这个独木桥上面动的人啦！接下来，来看一看 MAC 这个讯框的内容吧！

前导码	目的地址	来源地址	数据字段通讯	主要数据	检查码
8 Bytes	6 Bytes	6 Bytes	2 Bytes	46-1500 Bytes	4 Bytes

图 2.2-3、以太网络的 MAC 讯框

上图中的目的地址与来源地址指的就是网卡卡号 (hardware address, 硬件地址)，我们前面提到，每一张网卡都有一个独一无二的卡号， 那个卡号的目的就在这个讯框的表头数据使用到啦！硬件地址最小由 00:00:00:00:00:00 到 FF:FF:FF:FF:FF:FF (16 进位法)， 这 6 bytes 当中，前 3bytes 为厂商的代码，后 3bytes 则是该厂商自行设定的装置码了。

在 Linux 当中，你可以使用 ifconfig 这个指令来查阅你的网络卡卡号喔！特别注意，在这个 MAC 的传送中，他仅在局域网络内生效，如果跨过不同的网域 (这个后面 IP 的部分时会介绍)，那么来源与目的的硬件地址就会跟着改变了。 这是因为变成不同网络卡之间的交流了嘛！所以卡号当然不同了！如下所示：

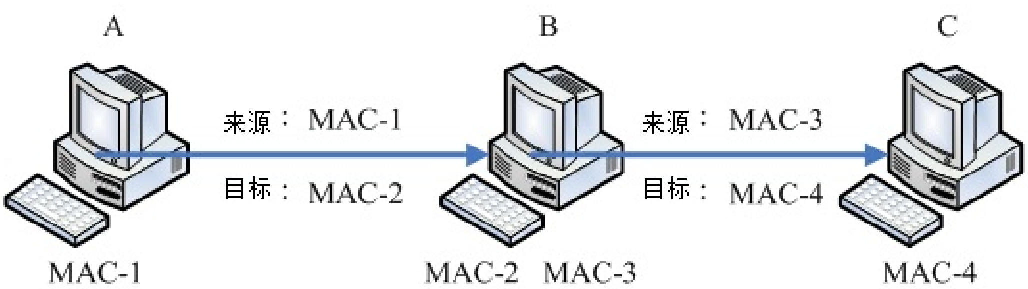


图 2.2-4、同一讯框在不同网域的主机间传送时，讯框的表头变化

例如上面的图标，我的数据要由计算机 A 通过 B 后才送达 C ，而 B 计算机有两块网络卡，其中 MAC-2 与 A 计算机的 MAC-1 互通，至于 MAC-3 则与 C 计算机的 MAC-4 互通。但是 MAC-1 不能与 MAC-3 与 MAC-4 互通，为啥？因为 MAC-1 这块网络卡并没有与 MAC-3 及

MAC-4 使用同样的 switch/hub 相接嘛！所以，数据的流通会变成：

1. 先由 MAC-1 传送到 MAC-2 ，此时来源是 MAC-1 而目的地是 MAC-2；
2. B 计算机接收后，察看该讯框，发现目标其实是 C 计算机，而为了与 C 计算机沟通， 所以他会将讯框内的来源 MAC 改为 MAC-3 ，而目的改为 MAC-4 ，如此就可以直接传送到 C 计算机了。

也就是说，只要透过 B (就是路由器) 才将封包送到另一个网域 (IP 部分会讲) 去的时候， 那么讯框内的硬件地址就会被改变，然后才能够在同一个网域里面直接进行讯框的流通啊！

Tips: 由于网络卡卡号是跟着网络卡走的，并不会因为重灌操作系统而改变， 所以防火墙软件大多也能够针对网络卡来进行抵挡的工作喔！ 不过抵挡网卡仅能在局域网络内进行而已，因为 MAC 不能跨 router 嘛！！



- 为什么资料量最小要 46 最大为 1500 bytes 呢？

  讯框内的数据内容最大可达 1500bytes 这我们现在知道了，那为何要规范最小数据为 46bytes 呢？这是由于 CSMA/CD 机制所算出来的！ 在这个机制上面可算出若要侦测碰撞，则讯框总数据量最小得要有 64bytes ，那再扣除目的地址、来源地址、检查码 (前导码不算) 后， 就可得到数据量最小得要有 46bytes 了！也就是说，如果妳要传输的数据小于 46byes ，那我们的系统会主动的填上一些填充码， 以补齐至少 46bytes 的容量才行！

2.2.5 MTU 最大传输单位

通过上面 MAC 封装的定义，现在我们知道标准以太网络讯框所能传送的数据量最大可以到达 1500 bytes ， 这个数值就被我们称为 MTU (Maximum Transmission Unit, 最大传输单位)。 你得要注意的是，每种网络接口的 MTU 都不相同，因此有的时候在某些网络文章上面你会看到 1492 bytes 的 MTU 等等。不过，在以太网络上，标准的定义就是 1500 bytes。

在待会儿会介绍到的 IP 封包中，这个 IP 封包最大可以到 65535 bytes，比 MTU 还要大呢！ 既然礼物 (IP) 都比盒子 (MAC) 大，那怎么可能放的进去啊？所以啰， IP 封包是可以进行拆解的，然后才能放到 MAC 当中啊！等到数据都传到目的地， 再由目的地的主机将他组装回来就是了。所以啰，如果 MTU 能够大一些的话，那么 IP 封包的拆解情况就会降低， 封包与封包传送之间的等待时间 (前一小节提到的 96 bit time) 也会减少，就能够增加网络带宽的使用啰！

为了这个目的，所以 Gigabit 的以太网络媒体才有支持 Jumbo frame 的嘛！这个 Jumbo frame 一般都定义到 9000bytes。 那你会说，既然如此，我们的 MTU 能不能改成 9000bytes 呢？这样一来不就能够减少数据封包的拆解，以增加网络使用率吗？ 是这样没错，而且，你也确实可以在 Linux 系统上更改 MTU 的！但是，如果考虑到整个网络，那么我们不建议你修改这个数值。 为什么呢？

我们的封包总是需要在 Internet 上面跑吧？你无法确认所有的网络媒体都是支持那么大的 MTU 对吧！ 如果你的 9000 bytes 封包通过一个不支持 Jumbo frame 的网络媒体时，好一点的是该网络媒体 (例如 switch/router 等) 会主动的帮你重组而进行传送，差一点的可能就直接回报这个封包无效而丢弃了～这个时候可就糗大啰～ 所以， MTU 设定为 9000 这种事情，大概仅能在内部网络的环境中作～举例来说，很多的内部丛集系统 (cluster) 就将他们的内部网络环境 MTU 设定为 9000，但是对外的适配卡可还是原本的标准 1500 喔！ ^_^

也就是说，不论你的网络媒体支持 MTU 到多大，你必须要考虑到你的封包需要传到目的地时， 所需要经过的所有网络媒体，然后再来决定你的 MTU 设定才行。就因为这样，我们才不建议你修改标准以太网络的 MTU 嘛！

Tips: 早期某些网络媒体 (例如 IP 分享器) 支持的是 802.2, 802.3 标准所组合成的 MAC 封装，它的 MTU 就是 1492 ， 而且这些设备可能不会进行封包重组，因此早期网络上面常常有朋友问说，他们连上某些网站时，总是会联机逾时而断线。 但透过修改客户端的 MTU 成为 1492 之后，上网就没有问题了。原因是什么呢？读完上头的数据，您应该能理解了吧？^_^



2.2.6 集线器、交换器与相关机制

- 共不共享很重要，集线器还是交换器？ (注11)

  刚刚我们上面提到了，当一个很忙碌的网络在运作时，集线器 (hub) 这个网络共享媒体就可能会发生碰撞的情况， 这是因为 CSMA/CD 的缘故。那有没有办法避免这种莫名其妙的封包碰撞情况呢？有的，那就使用非共享媒体的交换器即可啊！

  交换器 (switch) 等级非常多，我们这里仅探讨支持 OSI 第二层的交换器。交换器与集线器最大的差异，在于交换器内有一个特别的内存， 这个内存可以记录每个 switch port 与其连接的 PC 的 MAC 地址，所以，当来自 switch 两端的 PC 要互传数据时，每个讯框将直接透过交换器的内存数据而传送到目标主机上！ 所以 switch 不是共享媒体，且 switch 的每个埠口 (port) 都具有独立的带宽喔！

举例来说，10/100 的 Hub 上链接 5 部主机，那么整个 10/100Mbps 是分给这五部主机的， 所以这五部主机总共只能使用 10/100Mbps 而已。那如果是 switch 呢？由于『每个 port 都具有 10/100Mbps 的带宽』， 所以就看你当时的传输行为是如何啰！举例来说，如果是底下的状况时，每个联机都是 10/100 Mbps 的。

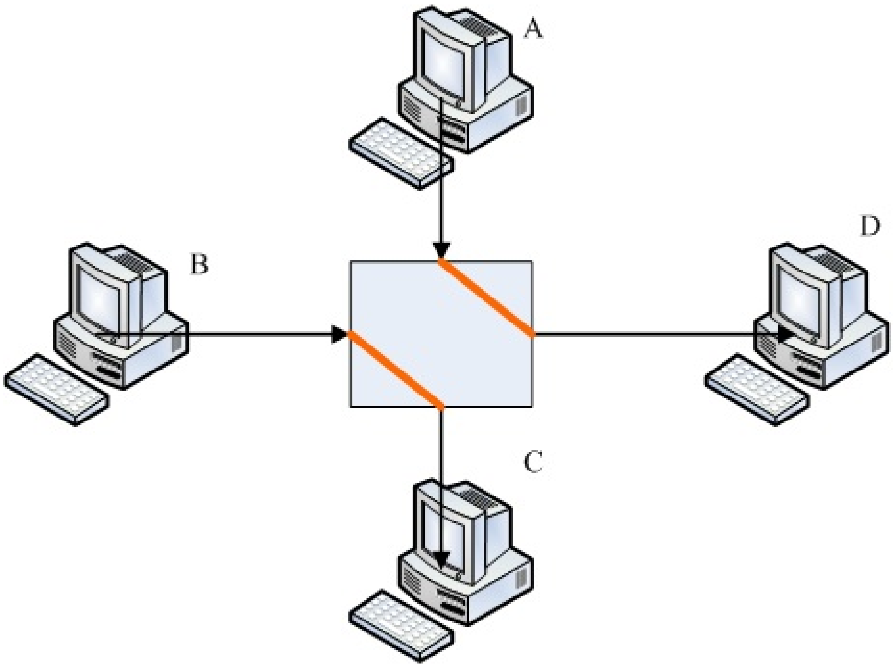


图 2.2-5、交换器每个埠口的带宽使用示意图

A 传送到 D 与 B 传送到 C 都独自拥有 10/100Mbps 的带宽，两边并不会互相影响！ 不过，如果是 A 与 D 都传给 C 时，由于 C port 就仅有 10/100Mbps ，等于 A 与 D 都需要抢 C 节点的 10/100Mbps 来用的意思。 总之，你就是得要记得的是，switch 已经克服了封包碰撞的问题，因为他有个 switch port 对应 MAC 的相关功能， 所以 switch 并非共享媒体喔！同时需要记得的是，现在的 switch 规格很多， 在选购的时候，千万记得选购可以支持全双工/半双工，以及支持 Jumbo frame 的为佳！

- 什么是全双工/半双工(full-duplex, half-duplex)

前面谈到网络线时，我们知道八蕊的网络线实际上仅有两对被使用，一对是用在传送，另一对则是在接收。 如果两端的 PC 同时支持全双工时，那表示 Input/Output 均可达到 10/100Mbps， 亦即数据的传送与接收同时均可达到 10/100bps 的意思，总带宽则可达到 20/200Mbps 啰 (其实是有点语病的，因为 Input 可达 10/100Mbps， output 可达 10/100Mbps ， 而不是 Input 可直接达到 20/200Mbps 喔！)如果你的网络环境想要达到全双工时， 使用共享媒体的 Hub 是不可能的，因为网络线脚位的关系，无法使用共享媒体来达到全双工的！ 如果你的 switch 也支持全双工模式，那么在 switch 两端的 PC 才能达到全双工喔！

- 自动协调速度机制 (auto-negotiation)：

我们都知道现在的以太网络卡是可以向下支持的，亦即是 Gigabit 网络卡可以与早期的 10/100Mbps 网络卡链接而不会发生问题。但是，此时的网络速度是怎样判定呢？ 早期的 switch/hub 必须要手动切换速度才行，新的 hub/switch 因为有支持 auto-negotiation 又称为 N-Way 的功能，他可自动的协调出最高的传输速度来沟通喔！如果有 Gigabit 与 10/100Mbps 在 switch 上面， 则 N-Way 会先使用最高的速度 (gigabit) 测试是否能够全部支持，如果不行的话，就降速到下一个等级亦即 100 Mbps 的速度来运作的！

- 自动分辨网络线跳线或并行线 (Auto MDI/MDIX)：

那么我们是否需要自行分辨并行线与跳线呢？不需要啦！因为 switch 若含有auto MDI/MDIX 的功能时， 会自动分辨网络线的脚位来调整联机的，所以你就不需要管你的网络线是跳线还是并行线啰！方便吧！ ^_^

- 讯号衰减造成的问题

由于电子讯号是会衰减的，所以当网络线过长导致电子讯号衰减的情况严重时， 就会导致联机质量的不良了。因此，链接各个节点的网络线长度是有限制的喔！ 不过，一般来说，现今的以太网络 CAT5 等级的网络线大概都可以支持到 100 公尺的长度， 所以应该无庸担心才是呐！

但是，造成讯号衰减的情况并非仅有网络线长度而已！如果你的网络线折得太严重(例如在门边常常被门板压，导致变形) ，或者是自行压制网络线接头，但是接头部分的八蕊蕊线缠绕度不足导致电磁干扰严重， 或者是网络线放在户外风吹日晒导致脆化的情况等等，都会导致电子讯号传递的不良而造成联机质量恶劣， 此时常常就会发现偶而可以联机、有时却又无法联机的问题了！因此，当你需要针对企业内部来架设整体的网络时， 注意结构化布线可是很重要的喔！

- 结构化布线

所谓的结构化布线指的是将各个网络的组件分别拆开，分别安装与布置到企业内部， 则未来想要提升网络硬件等级或者是移动某些网络设备时，只需要更动类似配线盘的机柜处， 以及末端的墙上预留孔与主机设备的联机就能够达到目的了。例如底下的图示：

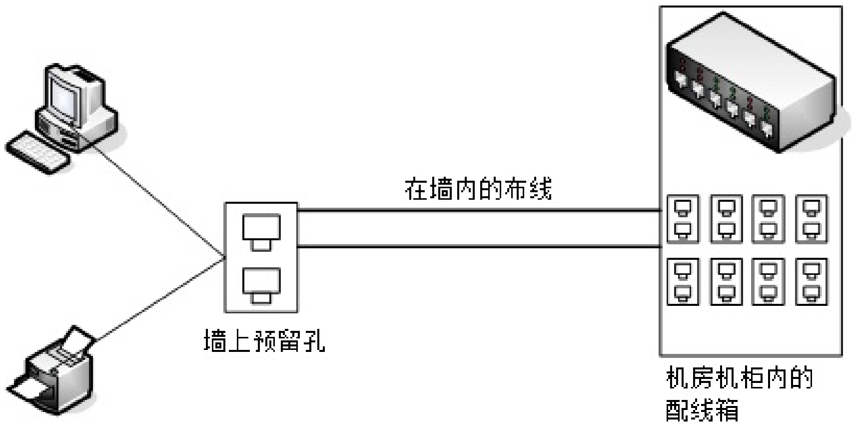


图 2.2-6、结构化布线简易图标

在墙内的布线需要很注意，因为可能一布线完成后就使用 5-10 年以上喔！那你需要注意的仅有末端墙上的预留孔以及配线端部分。 事实上，光是结构化布线所需要选择的网络媒体与网络线的等级，还有机柜、机架，以及美化与隐藏网络线的材料等等的挑选， 以及实际施工所需要注意的事项，还有所有硬件、施工所需要注意的标准规范等等， 已经可以写满厚厚一本书，而鸟哥这里的文章旨在介绍一个中小企业内部主机数量较少的环境， 所以仅提到最简单的以一个或两个交换器 (swtich) 串接所有网络设备的小型星形联机状态而已。

如果你有需要相关硬件结构化布线的信息，可以参考风信子兄翻译的『Swtich and Fast 以太网络』一书的后半段！至于网络上的高手吗？你可以前往酷学园请教 ZMAN (http://http://wordpress.morezman.com/) 大哥喔！

2.3 TCP/IP 的网络层相关封包与数据

我们现在知道要有网络的话，必须要有网络相关的硬件，而目前最常见的网络硬件接口为以太网络，包括网络线、网络卡、Hub/Switch 等等。而以太网络上面的传输使用网络卡卡号为基准的 MAC 讯框，配合 CSMA/CD 的标准来传送讯框，这就是硬件部分。在软件部分，我们知道 Internet 其实就是 TCP/IP 这个通讯协议的通称，Internet 是由 InterNIC(注12) 所统一管理的， 但其实他仅是负责分配 Internet 上面的 IP 以及提供相关的 TCP/IP 技术文件而已。不过 Internet 最重要的就是 IP 啊！所以， 这个小节就让我们来讲讲网络层的 IP 与路由吧！

2.3.1 IP 封包的封装

目前因特网社会的 IP 有两种版本，一种是目前使用最广泛的 IPv4 (Internet Protocol version 4, 因特网协定第四版)， 一种则是预期未来会热门的 IPv6 。IPv4 记录的地址由于仅有 32 位，预计在 2020 年前后就会分发完毕，如此一来， 新兴国家或者是新的网络公司，将没有网络可以使用。为了避免这个问题发生，因此就有 IPv6 的产生。 IPv6 的地址可以达到 128 位，可以多出 2 的 96 次方倍的网址数量，这样的 IP 数量几乎用不完啦！虽然 IPv6 具有前瞻性，但目前主流媒体大多还是使用 IPv4 ，因此本文主要谈到的 IP 都指 IPv4 而言喔！(注13)

我们在前一小节谈到 MAC 的封装，那么 IP 封包的封装也得要来了解一下，才能知道 IP 到底是如何产生的啊！ IP 封包可以达到 65535 bytes 这么大，在比 MAC 大的情况下，我们的操作系统会对 IP 进行拆解的动作。至于 IP 封装的表头数据绘制如下：(下图第一行为每个字段的 bit 数)

4 bits	4 bits	8 bits	3 bits	13 bits
Version	IHL	Type of Service	Total Length	
Identification	Flags	Fragmentation Offset		
Time To Live	Protocol	Header Checksum		
Source Address				
Destination Address				
Options	Padding			
Data				

图 2.3-1、IP 封包的表头资料

在上面的图示中有个地方要注意，那就是『每一行所占用的位数为 32 bits』， 各个表头的内容分别介绍如下：

- Version(版本) 宣告这个 IP 封包的版本，例如目前惯用的还是 IPv4 这个版本就在这里宣告。

- IHL(Internet Header Length, IP表头的长度) 告知这个 IP 封包的表头长度，使用的单位应该是字组 (word) ，一个字组为 4bytes 大小喔。

- Type of Service(服务类型) 这个项目的内容为『PPPDTRUU』，表示这个 IP 封包的服务类型，主要分为： PPP：表示此 IP 封包的优先度，目前很少使用； D：若为 0 表示一般延迟(delay)，若为 1 表示为低延迟； T：若为 0 表示为一般传输量 (throughput)，若为 1 表示为高传输量； R：若为 0 表示为一般可靠度(reliability)，若为 1 表示高可靠度。 UU：保留尚未被使用。 举例来说，gigabit 以太网络的种种相关规格可以让这个 IP 封包加速且降低延迟，某些特殊的标志就是在这里说明的。

- Total Length(总长度) 指这个 IP 封包的总容量，包括表头与内容 (Data) 部分。最大可达 65535 bytes。

- Identification(辨别码) 我们前面提到 IP 袋子必须要放在 MAC 袋子当中。不过，如果 IP 袋子太大的话，就得先要将 IP 再重组成较小的袋子然后再放到 MAC 当中。而当 IP 被重组时，每个来自同一个 IP 的小袋子就得要有个标识符以告知接收端这些小袋子其实是来自同一个 IP 封包才行。 也就是说，假如 IP 封包其实是 65536 那么大 (前一个 Total Length 有规定)， 那么这个 IP 就得要再被分成更小的 IP 分段后才能塞进 MAC 讯框中。那么每个小 IP 分段是否来自同一个 IP 资料，呵呵！那就是这个标识符的功用啦！

- Flags(特殊旗标) 这个地方的内容为『0DM』，其意义为： D：若为 0 表示可以分段，若为 1 表示不可分段 M：若为 0 表示此 IP 为最后分段，若为 1 表示非最后分段。

- Fragment Offset(分段偏移) 表示目前这个 IP 分段在原始的 IP 封包中所占的位置。就有点像是序号啦，有这个序号才能将所有的小 IP 分段组合成为原本的 IP 封包大小嘛！透过 Total Length, Identification, Flags 以及这个 Fragment Offset 就能够将小 IP 分段在收受端组合起来啰！

- Time To Live(TTL, 存活时间) 表示这个 IP 封包的存活时间，范围为 0-255。当这个 IP 封包通过一个路由器时， TTL 就会减一，当 TTL 为 0 时，这个封包将会被直接丢弃。说实在的，要让 IP 封包通过 255 个路由器，还挺难的～ ^_^

- Protocol Number(协定代码) 来自传输层与网络层本身的其他数据都是放置在 IP 封包当中的，我们可以在 IP 表头记载这个 IP 封包内的资料是啥， 在这个字段就是记载每种数据封包的内容啦！在这个字段记载的代码与相关的封包协议名称如下所示：

```
| IP 内的号码 | 封包协议名称(全名) |
| --- | --- |
| 1 | ICMP (Internet Control Message Protocol) |
| 2 | IGMP (Internet Group Management Protocol) |
| 3 | GGP (Gateway-to-Gateway Protocol) |
| 4 | IP (IP in IP encapsulation) |
| 6 | TCP (Transmission Control Protocol) |
| 8 | EGP (Exterior Gateway Protocol) |
| 17 | UDP (User Datagram Protocol) |
```

当然啦，我们比较常见到的还是那个 TCP, UDP, ICMP 说！

- Header Checksum(表头检查码) 用来检查这个 IP 表头的错误检验之用。
- Source Address 还用讲吗？当然是来源的 IP 地址，从这里我们也知道 IP 是 32 位喔！
- Destination Address 有来源还需要有目标才能传送，这里就是目标的 IP 地址。
- Options (其他参数) 这个是额外的功能，提供包括安全处理机制、路由纪录、时间戳、严格与宽松之来源路由等。
- Padding(补齐项目) 由于 Options 的内容不一定有多大，但是我们知道 IP 每个数据都必须要是 32 bits，所以，若 Options 的数据不足 32 bits 时，则由 padding 主动补齐。

你只要知道 IP 表头里面含有： TTL, Protocol, 来源地址与目标地址也就够了！而这个 IP 表头的来源与目标 IP ，以及那个判断通过多少路由器的 TTL ，就能了解到这个 IP 将被如何传送到目的端呐。后续各小节我们将介绍 IP 的组成与范围，还有 IP 封包如何传送的机制 (路由) 等等。

2.3.2 IP 地址的组成与分级

现在我们知道 IP (Internet Protocol) 其实是一种网络封包，而这个封包的表头最重要的就是那个 32 位的来源与目标地址！ 为了方便记忆，所以我们也称这个 32 bits 的数值为 IP 网络地址就是了。因为网络是人类发明的，所以很多概念与邮务系统类似！ 那这个 IP 其实就类似所谓的『门牌号码』啦！那么这个 IP 有哪些重要的地方需要了解的呢？底下我们就来谈一谈吧！

既然 IP 的组成是 32 bits 的数值，也就是由 32 个 0 与 1 组成的一连串数字！那么当我们思考所有跟 IP 有关的参数时，你就应该要将该参数想成是 32 位的数据喔！ 不过，因为人类对于二进制实在是不怎么熟悉，所以为了顺应人们对于十进制的依赖性，因此，就将 32 bits 的 IP 分成四小段，每段含有 8 个 bits ，将 8 个 bits 计算成为十进制，并且每一段中间以小数点隔开，那就成了目前大家所熟悉的 IP 的书写模样了。如下所示：

```
IP 的表示式：
00000000.00000000.00000000.00000000   ==&gt; 0.0.0.0
11111111.11111111.11111111.11111111   ==&gt; 255.255.255.255
```

所以 IP 最小可以由 0.0.0.0 一直到 255.255.255.255 哩！但在这一串数字中，其实还可以分为两个部分喔！ 主要分为 Net_ID (网域号码)与 Host_ID (主机号码) 两部份。我们先以 192.168.0.0 ~ 192.168.0.255 这个 Class C 的网域当作例子来说明好了：

```
192.168.0.0~192.168.0.255 这个 Class C 的说明：
11000000.10101000.00000000.00000000
11000000.10101000.00000000.11111111
&#124;----------Net_ID---------&#124;-host--&#124;
```

在上面的范例当中，前面三组数字 (192.168.0) 就是网域号码，最后面一组数字则称为主机号码。 至于同一个网域的定义是『在同一个物理网段内，主机的 IP 具有相同的 Net_ID ，并且具有独特的 Host_ID』，那么这些 IP 群就是同一个网域内的 IP 网段啦！

Tips: 什么是物理网段呢？当所有的主机都是使用同一个网络媒体串在一起， 这个时候这些主机在实体装置上面其实是联机在一起的，那么就可以称为这些主机在同一个物理网段内了！ 同时并请注意，同一个物理网段之内，可以依据不同的 IP 的设定，而设定成多个『IP 网段』喔！



上面例子当中的 192.168.0.0, 192.168.0.1, 192.168.0.2, ..., 192.168.0.255 (共 256 个) 这些 IP 就是同一个网域内的 IP 群(同一个网域也称为同一个网段！)，请注意，同一个 Net_ID 内，不能具有相同的 Host_ID ，否则就会发生 IP 冲突，可能会造成两部主机都没有办法使用网络的问题！

- IP 在同一网域的意义

那么同一个网域该怎么设定，与将 IP 设定在同一个网域之内有什么好处呢？

- Net_ID 与 Host_ID 的限制： 在同一个网段内，Net_ID 是不变的，而 Host_ID 则是不可重复，此外，Host_ID 在二进制的表示法当中，不可同时为 0 也不可同时为 1 ，因为全为 0 表示整个网段的地址 (Network IP)，而全为 1 则表示为广播的地址 (Broadcast IP)。例如上面的例子当中，192.168.0.0 (Host_ID 全部为 0)以及 192.168.0.255 (Host_ID 全部为 1) 不可用来作为网段内主机的 IP 设定，也就是说，这个网段内可用来设定主机的 IP 是由 192.168.0.1 到 192.168.0.254；

- 在区网内透过 IP 广播传递数据 在同物理网段的主机如果设定相同的网域 IP 范围 (不可重复)，则这些主机都可以透过 CSMA/CD 的功能直接在区网内用广播进行网络的联机，亦即可以直接网卡对网卡传递数据 (透过 MAC 讯框)；

- 设定不同区网在同物理网段的情况 在同一个物理网段之内，如果两部主机设定成不同的 IP 网段，则由于广播地址的不同，导致无法透过广播的方式来进行联机。 此时得要透过路由器 (router) 来进行沟通才能将两个网域连结在一起。
- 网域的大小 当 Host_ID 所占用的位越大，亦即 Host_ID 数量越多时，表示同一个网域内可用以设定主机的 IP 数量越多。

所以说，贵单位公司内的计算机群，或者是你宿舍或家里面的所有计算机，当然都设定在同一个网域内是最方便的， 因为如此一来每一部计算机都可以直接透过 MAC 来进行数据的交流，而不必经由 Router (路由器) 来进行封包的转递呢！(Router 这部份在第八章才会提及)。

- IP 与门牌号码的联想

刚接触到 IP 组成的朋友都很困扰，又分啥网域号码与主机号码，烦死了！其实，你不用烦恼啊！使用门牌号码的概念来想即可。 既然 IP 是门牌，那拿我们昆山科技大学的门牌来说好了，我们的门牌是：『台南市永康区大湾路 949 号』， 假设整个大湾路是同一个巷弄，那么我们这个门牌的网域号码『台南市永康区大湾路』而我的主机号码就是『949 号』， 那么整条大湾路上面只要是开头为『台南市永康区大湾路』的，就是跟我们同一个网域啰！当然啦，门牌号码不可能有第二个 949 号啊！这样理解否？

另外，Host_ID 全为 0 与全为 1 (二进制的概念) 时，代表整条巷子的第一个与最后一个门牌，而第一个门牌我们让他代表整条巷子， 所以又称为 Network IP，就是巷子口那个 XXX 巷的立牌啦！至于最后一个 IP ，则代表巷子尾，亦即本条巷子的最后一个门牌， 那就是我们在巷子内广播时的最后一个 IP ，又称为 Broadcast IP 的啰。

在我们这个巷子内，我们可以透过大声公用广播的方式跟大家沟通讯息，例如前几年很热门的张君雅小妹妹的泡面广告， 在巷子内透过广播告诉张君雅小妹妹，你阿嬷将泡面煮好了，赶快回家吃面去！那如果不是张君雅小妹妹呢？就将该讯息略过啊！ 这样有没有联想到 CSMA/CD 的概念呢？

那如果你的数据不是要给本巷子内的门牌呢？此时你就得要将资料拿给巷子内的邮局 (路由器)，由邮局帮你传送， 你只要知道巷子内的那间邮局在哪里即可，其他的就让邮局自己帮你把信件传出去即可啊！这就是整个区网与门牌对应的想法！ 这样有没有比较清晰啊？

- IP 的分级

你应该要想到一个问题，那就是我的总门牌『台南市永康区大湾路 949 号』中，到哪里是巷子而到哪里是门牌？ 如果到『台南市』是巷子，那么我的门牌将有好多乡镇的组成，如果巷子号码到『台南市永康区』时， 那么我们的门牌就又少了点。所以说，这个『巷子』的大小，将会影响到我们主机号码的数量！

为了解决这个问题，以及为了 IP 管理与发放注册的方便性，InterNIC 将整个 IP 网段分为五种等级， 每种等级的范围主要与 IP 那 32 bits 数值的前面几个位有关，基本定义如下：

```
以二进制说明 Network 第一个数字的定义：
Class A : 0xxxxxxx.xxxxxxxx.xxxxxxxx.xxxxxxxx  ==&gt; NetI_D 的开头是 0
          &#124;--net--&#124;---------host------------&#124;
Class B : 10xxxxxx.xxxxxxxx.xxxxxxxx.xxxxxxxx  ==&gt; NetI_D 的开头是 10
          &#124;------net-------&#124;------host------&#124;
Class C : 110xxxxx.xxxxxxxx.xxxxxxxx.xxxxxxxx  ==&gt; NetI_D 的开头是 110
          &#124;-----------net-----------&#124;-host--&#124;
Class D : 1110xxxx.xxxxxxxx.xxxxxxxx.xxxxxxxx  ==&gt; NetI_D 的开头是 1110
Class E : 1111xxxx.xxxxxxxx.xxxxxxxx.xxxxxxxx  ==&gt; NetI_D 的开头是 1111

五种分级在十进制的表示：
Class A :   0.xx.xx.xx ~ 127.xx.xx.xx
Class B : 128.xx.xx.xx ~ 191.xx.xx.xx
Class C : 192.xx.xx.xx ~ 223.xx.xx.xx
Class D : 224.xx.xx.xx ~ 239.xx.xx.xx
Class E : 240.xx.xx.xx ~ 255.xx.xx.xx
```

根据上表的说明，我们可以知道，你只要知道 IP 的第一个十进制数，就能够约略了解到该 IP 属于哪一个等级， 以及同网域 IP 数量有多少。这也是为啥我们上头选了 192.168.0.0 这一 IP 网段来说明时，会将巷子定义到第三个数字之故。 不过，上表中你只要记忆三种等级，亦即是 Class A, B, C 即可，因为 Class D 是用来作为群播 (multicast) 的特殊功能之用 (最常用在大批计算机的网络还原)，至于 Class E 则是保留没有使用的网段。因此，能够用来设定在一般系统上面的，就只有 Class A, B, C 三种等级的 IP 啰！

2.3.3 IP 的种类与取得方式

接下来要跟大家谈一谈也是很容易造成大家困扰的一个部分，那就是 IP 的种类！很多朋友常常听到什么『真实IP, 实体 IP, 虚拟 IP, 假的 IP....』烦都烦死了～其实不要太紧张啦！实际上，在 IPv4 里面就只有两种 IP 的类别，分别是：

- Public IP : 公共 IP ，经由 INTERNIC 所统一规划的 IP，有这种 IP 才可以连上 Internet ；
- Private IP : 私有 IP 或保留 IP，不能直接连上 Internet 的 IP ， 主要用于局域网络内的主机联机规划。

早在 IPv4 规划的时候就担心 IP 会有不足的情况，而且为了应付某些企业内部的网络设定，于是就有了私有 IP (Private IP) 的产生了。私有 IP 也分别在 A, B, C 三个 Class 当中各保留一段作为私有 IP 网段，那就是：

- Class A：10.0.0.0 - 10.255.255.255
- Class B：172.16.0.0 - 172.31.255.255
- Class C：192.168.0.0 - 192.168.255.255

由于这三段 Class 的 IP 是预留使用的，所以并不能直接作为 Internet 上面的连接之用，不然的话，到处就都有相同的 IP 啰！那怎么行！网络岂不混乱？所以啰，这三个 IP 网段就只做为内部私有网域的 IP 沟通之用。简单的说，他有底下的几个限制：

- 私有 IP 的路由信息不能对外散播 (只能存在内部网络)；
- 使用私有 IP 作为来源或目的地址的封包，不能透过 Internet 来转送 (不然网络会混乱)；
- 关于私有 IP 的参考纪录(如 DNS)，只能限于内部网络使用 (一样的原理啦)

这个私有 IP 有什么好处呢？由于他的私有路由不能对外直接提供信息，所以，你的内部网络将不会直接被 Internet 上面的 Cracker 所攻击！但是，你也就无法以私有 IP 来『直接上网』啰！因此相当适合一些尚未具有 Public IP 的企业内部用来规划其网络之设定！否则当你随便指定一些可能是 Public IP 的网段来规划你企业内部的网络设定时，万一哪一天真的连上 Internet 了，那么岂不是可能会造成跟 Internet 上面的 Public IP 相同了吗？

此外，在没有可用的公开网络情况下，如果你想要跟同学玩联机游戏怎办？也就是说，在区网内自己玩自己的联机游戏， 此时你只要规范好所有同学在同一段私有 IP 网段中，就能够顺利的玩你的网络啦！就这么简单呢！

那么万一你又要将这些私有 IP 送上 Internet 呢？这个简单，设定一个简单的防火墙加上 NAT (Network Address Transfer) 服务，你就可以透过 IP 伪装 (不要急，这个在后面也会提到) 来使你的私有 IP 的计算机也可以连上 Internet 啰！

- 特殊的 loopback IP 网段

好了，那么除了这个预留的 IP 网段的问题之外，还有没有什么其他的怪东西呢？当然是有啦！不然鸟哥干嘛花时间来唬 XX 呢？没错，还有一个奇怪的 Class A 的网域，那就是 lo 这个奇怪的网域啦 (注意：是小写的 o 而不是零喔)！这个 lo 的网络是当初被用来作为测试操作系统内部循环所用的一个网域，同时也能够提供给系统内部原本就需要使用网络接口的服务 (daemon) 所使用。

简单的说，如果你没有安装网络卡在的机器上面， 但是你又希望可以测试一下在你的机器上面设定的服务器环境到底可不可以顺利运作，这个时候怎么办， 嘿嘿！就是利用这个所谓的内部循环网络啦！这个网段在 127.0.0.0/8 这个 Class A，而且默认的主机 (localhost) 的 IP 是 127.0.0.1 呦！所以啰，当你启动了你的 WWW 服务器，然后在你的主机的 X-Window 上面执行 http://localhost 就可以直接看到你的主页啰！而且不需要安装网络卡呢！测试很方便吧！

此外，你的内部使用的 mail 怎么运送邮件呢？例如你的主机系统如何 mail 给 root 这个人呢？嘿嘿！也就是使用这一个内部循环啦！当要测试你的 TCP/IP 封包与状态是否正常时，可以使用这个呦！(所以哪一天有人问你嘿！你的主机上面没有网络卡， 那么你可以测试你的 WWW 服务器设定是否正确吗？这个时候可得回答：当然可以啰！使用 127.0.0.1 这个 Address 呀！ ^_^)

- IP 的取得方式

谈完了 IP 的种类与等级还有相关的子域概念后，接下来我们得来了解一下，那么主机的 IP 是如何设定的呢？ 基本上，主机的 IP 与相关网域的设定方式主要有：

- 直接手动设定(static)： 你可以直接向你的网管询问可用的 IP 相关参数，然后直接编辑配置文件 (或使用某些软件功能) 来设定你的网络。 常见于校园网络的环境中，以及向 ISP 申请固定 IP 的联机环境；

- 透过拨接取得： 向你的 ISP 申请注册，取得账号密码后，直接拨接到 ISP ，你的 ISP 会透过他们自己的设定，让你的操作系统取得正确的网络参数。 此时你并不需要手动去编辑与设定相关的网络参数啦。目前台湾的 ADSL 拨接、光纤到大楼、光纤到府等，大部分都是使用拨接的方式。 为因应用户的需求，某些 ISP 也提供很多不同的 IP 分配机制。包括 hinet, seednet 等等都有提供 ADSL 拨接后取得固定 IP 的方式喔！ 详情请向你的 ISP 洽询。

- 自动取得网络参数 (DHCP)： 在局域网络内会有一部主机负责管理所有计算机的网络参数，你的网络启动时就会主动向该服务器要求 IP 参数， 若取得网络相关参数后，你的主机就能够自行设定好所有服务器给你的网络参数了。最常使用于企业内部、IP 分享器后端、 校园网络与宿舍环境，及缆线宽带等联机方式。

不管是使用上面哪种方式取得的 IP ，你的 IP 都只有所谓的『 Public 与 Private IP 』而已！ 而其他什么浮动式、固定制、 动态式等等有的没有的，就只是告诉你这个 IP 取得的方式而已。举例来说，台湾地区 ADSL 拨接后取得的 IP 通常是 public IP， 但是鸟哥曾接到香港网友的来信，他们 ADSL 拨接后，取得的 IP 是 Private ，所以导致无法架设网站喔！

2.3.4 Netmask, 子网与 CIDR (Classless Interdomain Routing)

我们前面谈到 IP 是有等级的，而设定在一般计算机系统上面的则是 Class A, B, C。现在我们来想一想，如果我们设定一个区网， 使用的是 Class A ，那么我们很容易就会想到，哪有这么多计算机可以设定在同一个 Class A 的区段内 (256x256x256-2=16777214) ？ 而且，假设眞有这么多计算机好了，回想一下 CSMA/CD 吧，你的网络恐怕会一直非常停顿，因为妳得要接到一千多万台计算机对你的广播... 光是想到一千多万台的广播，你的网络还能使用吗？眞没效率！

此外，分为 Class 的 IP 等级，是为了管理方面的考虑，事实上，我们不可能将一个 Class A 仅划定为一个区网。举例来说， 我们昆山取得的 Public IP 是 120.xxx 开头的，但是其实我们只有 120.114.xxx.xxx 而已，并没有取得整个 Class A 喔！ 因为我们学校也用不了这么多嘛！这个时候，我们就得要理解一下啰，就是，怎么将 Class A 的网段变小？换句话说， 我们如何将网域切的更细呢？这样不就可以分出更多段的区网给大家设定了？

前面我们提到 IP 这个 32 位的数值中分为网域号码与主机号码，其中 Class C 的网域号码占了 24 位，而其实我们还可以将这样的网域切的更细，就是让第一个 Host_ID 被拿来作为 Net_ID ，所以，整个 Net_ID 就有 25 bits ，至于 Host_ID 则减少为 7 bits 。在这样的情况

下，原来的一个 Class C 的网域就可以被切分为两个子域，而每个子域就有『 256/2 - 2 = 126 』个可用的 IP 了！这样一来，就能够将原本的一个网域切为两个较细小的网域，方便分门别类的设计喔。

- Netmask, 或称为 Subnet mask (子网掩码)

那到底是什么参数来达成子网的切分呢？那就是 Netmask (子网掩码) 的用途啦！这个 Netmask 是用来定义出网域的最重要的一个参数了！不过他也最难理解了～@_@。为了帮助大家比较容易记忆住 Netmask 的设定依据，底下我们介绍一个比较容易记忆的方法。同样以 192.168.0.0 ~ 192.168.0.255 这个网域为范例好了，如下所示，这个 IP 网段可以分为 Net_ID 与 Host_ID，既然 Net_ID 是不可变的，那就假设他所占据的 bits 已经被用光了 (全部为 1)，而 Host_ID 是可变的，就将他想成是保留着 (全部为 0)，所以， Netmask 的表示就成为：

```
192.168.0.0~192.168.0.255 这个 C Class 的 Netmask 说明
第一个 IP： 11000000.10101000.00000000.00000000
最后一个  : 11000000.10101000.00000000.11111111 &#124;----------Net_ID---------&#124;-host-
Netmask   : 11111111.11111111.11111111.00000000  &lt;== Netmask 二进制
          :   255    .  255    .  255    .   0      &lt;== Netmask 十进制
特别注意喔，netmask 也是 32 位，在数值上，位于 Net_ID 的为 1 而 Host_ID 为 0
```

将他转成十进制的话，就成为『255.255.255.0』啦！ 这样记忆简单多了吧！照这样的记忆方法，那么 A, B, C Class 的 Netmask 表示就成为这样：

```
Class A, B, C 三个等级的 Netmask 表示方式：
Class A : 11111111.00000000.00000000.00000000 ==&gt; 255\.  0\.  0\.  0
Class B : 11111111.11111111.00000000.00000000 ==&gt; 255.255\.  0\.  0
Class C : 11111111.11111111.11111111.00000000 ==&gt; 255.255.255\.  0
```

所以说， 192.168.0.0 ~ 192.168.0.255 这个 Class C 的网域中，他的 Netmask 就是 255.255.255.0 ！再来，我们刚刚提到了当 Host_ID 全部为 0 以及全部为 1 的时后该 IP 是不可以使用的，因为 Host_ID 全部为 0 的时后，表示 IP 是该网段的 Network ，至于全部为 1 的时后就表示该网段最后一个 IP ，也称为 Broadcast ，所以说，在 192.168.0.0 ~ 192.168.0.255 这个 IP 网段里面的相关网络参数就有：

```
Netmask:   255.255.255.0   &lt;==网域定义中，最重要的参数
Network:   192.168.0.0     &lt;==第一个 IP
Broadcast: 192.168.0.255   &lt;==最后一个 IP
可用以设定成为主机的 IP 数：
192.168.0.1 ~ 192.168.0.254
```

- 子网切分

好了，刚刚提到 Class C 还可以继续进行子域 (Subnet) 的切分啊，以 192.168.0.0 ~192.168.0.255 这个情况为例，他要如何再细分为两个子域呢？我们已经知道 Host_ID 可以拿来当作 Net_ID，那么 Net_ID 使用了 25 bits 时，就会如下所示：

```
原本的 C Class 的 Net_ID 与 Host_ID 的分别
11000000.10101000.00000000.00000000      Network:   192.168.0.0
11000000.10101000.00000000.11111111      Broadcast: 192.168.0.255
&#124;----------Net_ID---------&#124;-host--&#124;

切成两个子网之后的 Net_ID 与 Host_ID 为何？
11000000.10101000.00000000.0 0000000  多了一个 Net_ID 了, 为 0 (第一个子网)
11000000.10101000.00000000.1 0000000  多了一个 Net_ID 了, 为 1 (第二个子网)
&#124;----------Net_ID-----------&#124;-host--&#124;

第一个子网
Network:   11000000.10101000.00000000.0 0000000   192.168.0.0
Broadcast: 11000000.10101000.00000000.0 1111111   192.168.0.127
           &#124;----------Net_ID-----------&#124;-host-&#124;
Netmask:   11111111.11111111.11111111.1 0000000   255.255.255.128

第二个子网
Network:   11000000.10101000.00000000.1 0000000   192.168.0.128
Broadcast: 11000000.10101000.00000000.1 1111111   192.168.0.255
           &#124;----------Net_ID-----------&#124;-host-&#124;
Netmask:   11111111.11111111.11111111.1 0000000   255.255.255.128
```

所以说，当再细分下去时，就会得到两个子域，而两个子域还可以再细分下去喔 (Net*ID 用掉 26 bits ....)*。呵呵！如果你真的能够理解 *IP, Network, Broadcast, Netmask* 的话，恭喜你，未来的服务器学习之路已经顺畅了一半啦！ ^^

例题：试着计算出 172.16.0.0，但 Net_ID 占用 23 个位时，这个网域的 Netmask, Network, Broadcast 等参数答：由于 172.16.xxx.xxx 是在 Class B 的等级当中，亦即 Net_ID 是 16 位才对。不过题目给的 Net_ID 占用了 23 个位喔！ 等于是向 Host_ID 借了 (23-16) 7 个位用在 Net_ID 当中。所以整个 IP 的地址会变成这样：

```
预设：        172  .  16    .0000000 0.00000000
          &#124;----Net_ID--------------&#124;--Host---&#124;
Network:     172  .  16    .0000000 0.00000000   172.16.0.0
Broadcast:   172  .  16    .0000000 1.11111111   172.16.1.255
Netmask:  11111111.11111111.1111111 0.00000000   255.255.254.0
```

鸟哥在这里有偷懒，因为这个 IP 段的前 16 个位不会被改变，所以并没有计算成二进制 (172.16)， 真是不好意思啊～至于粗体部分则是代表 host_ID 啊！

其实子网的计算是有偷吃步的，我们知道 IP 是二进制，每个位就是 2 的次方。又由于 IP 数量都是平均分配到每个子网去， 所以，如果我们以 192.168.0.0 ~ 192.168.0.255 这个网段来说，要是给予 Net_ID 是 26 位时，总共分为几段呢？ 因为 26-24=2 ，所以总共用掉两个位，因此有 2 的 2 次方，得到 4 个网段。再将 256 个 IP 平均分配到 4 个网段去， 那我们就可以知道这四个网段分别是：

- 192.168.0.0~192.168.0.63
- 192.168.0.64~192.168.0.127
- 192.168.0.128~192.168.0.191
- 192.168.0.192~192.168.0.255

有没有变简单的感觉啊？那你再想想，如果同样一个网段，那 Net_ID 变成 27 个位时，又该如何计算呢？自己算算看吧！

- 无层级 IP： CIDR (Classless Interdomain Routing)

一般来说，如果我们知道了 Network 以及 Netmask 之后，就可以定义出该网域的所有 IP 了！因为由 Netmask 就可以推算出来 Broadcast 的 IP 啊！因此，我们常常会以 Network 以及 Netmask 来表示一个网域，例如这样的写法：

```
Network/Netmask
192.168.0.0/255.255.255.0
192.168.0.0/24    &lt;==因为 Net_ID 共有 24 个 bits
```

另外，既然 Netmask 里面的 Net_ID 都是 1 ，那么 Class C 共有 24 bits 的 Net_ID ，所以啦，就有类似上面 192.168.0.0/24 这样的写法啰！这就是一般网域的表示方法。 同理可证，在上述的偷吃步计算网域方法中，四个网段的写法就可以写成：

- 192.168.0.0/26
- 192.168.0.64/26
- 192.168.0.128/26
- 192.168.0.192/26

事实上，由于网络细分的情况太严重，为了担心路由信息过于庞大导致网络效能不佳，因此，某些特殊情况下， 我们反而是将 Net_ID 借用来作为 Host_ID 的情况！这样就能够将多个网域写成一个啦！举例来说，我们将 256 个 Class C 的私有 IP (192.168.0.0~192.168.255.255) 写成一个路由信息的话，那么这个网段的写法就会变成：192.168.0.0/16，反而将 192 开头的 Class C 变成 class B 的样子了！ 这种打破原本 IP 代表等级的方式 (透过 Netmask 的规范) 就被称为无等级网域间路由 (CIDR) 啰！ (注14)

老实说，你无须理会啥是无等级网域间路由啦！只要知道，那个 Network/Netmask 的写法，通常就是 CIDR 的写法！ 然后，你也要知道如何透过 Netmask 去计算出 Network, Broadcast 及可用的 IP 等，那你的 IP 概念就相当完整了！^_^

2.3.5 路由概念

我们知道在同一个区网里面，可以透过 IP 广播的方式来达到资料传递的目的。但如果是非区网内的数据呢？ 这时就得要透过那个所谓的邮局 (路由器) 的帮忙了！这也是网络层非常重要的概念喔！先来看看什么是区网吧！

例题：请问 192.168.10.100/25 与 192.168.10.200/25 是否在同一个网域内？答：如果经过计算，会发现 192.168.10.100 的 Network 为 192.168.10.0 ，但是 192.168.10.200 的 Network 却是 192.168.10.128，由于 Net_ID 不相同，所以当然不在同一个网段内！ 关于 Network 与 Netmask 的算法则请参考上一小节。

如上题所述，那么这两个网段的数据无法透过广播来达到数据的传递啊，那怎办？ 此时就得要经过 IP 的路径选择 (routing) 功能啦！我们以下面图示的例子来做说明。 下列图示当中共有两个不同的网段，分别是 Network A 与 Network B，这两个网段是经由一部路由器 (Server A) 来进行数据转递的，好了，那么当 PC01 这部主机想要传送数据到 PC11 时， 他的 IP 封包该如何传输呢？

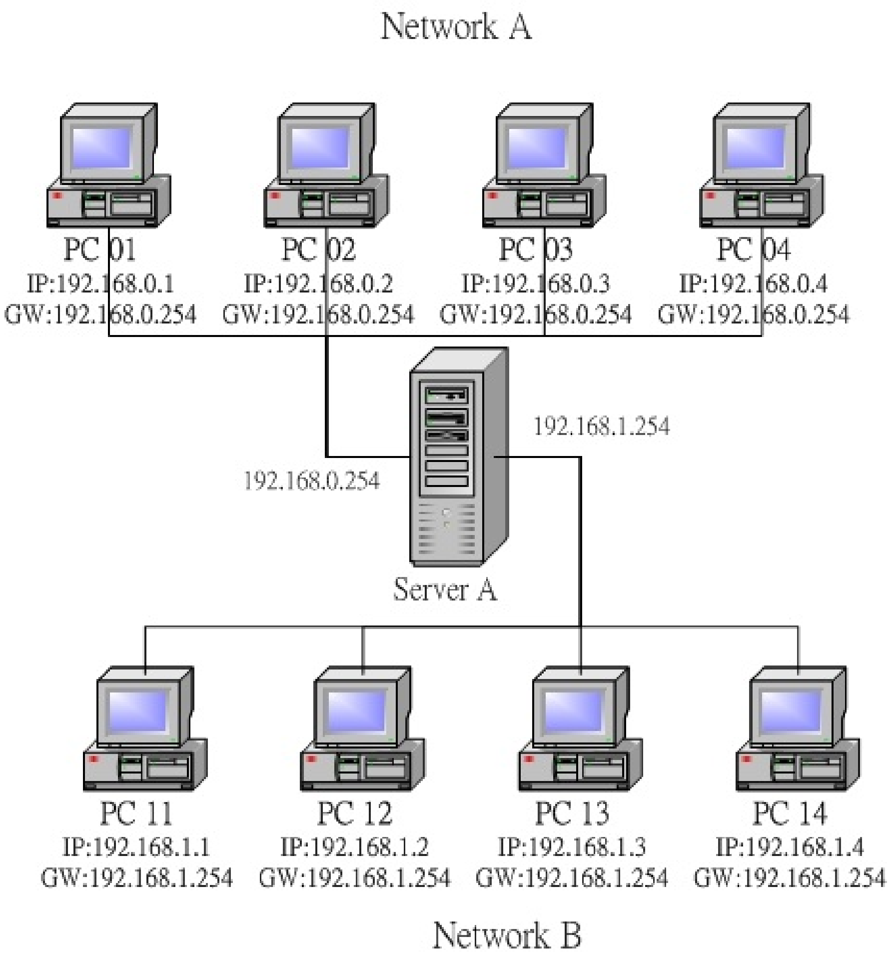


图 2.3-2、简易的路由示意图

我们知道 Network A(192.168.0.0/24) 与 Network B(192.168.1.0/24) 是不同网段，所以 PC01 与 PC11 是不能直接互通数据的。不过， PC01 与 PC11 是如何知道他们两个不在同一个网段内？这当然是透过 Net_ID 来发现的！那么当主机想要传送数据时，他主要的参考是啥？很简单！是『路由表 (route table)』，每部主机都有自己的路由表』， 让我们来看一看预设的情况下， PC01 要如何将数据传送到 PC02 呢？

1. 查询 IP 封包的目标 IP 地址： 当 PC01 有 IP 封包需要传送时，主机会查阅 IP 封包表头的目标 IP 地址；

2. 查询是否位于本机所在的网域之路由设定： PC01 主机会分析自己的路由表，当发现目标 IP 与本机 IP 的 Net_ID 相同时(同一网域)，则 PC01 会直接透过区网功能，将数据直接传送给目的地主机。

3. 查询预设路由 (default gateway)： 但在本案例中， PC01 与 PC11 并非同一网域，因此 PC01 会分析路由表当中是否有其他相符合的路由设定， 如果没有的话，就直接将该 IP 封包送到预设路由器 (default gateway) 上头去，在本案例当中 default gateway 则是 Server A 这一部。

4. 送出封包至 gateway 后，不理会封包流向： 当 IP 由 PC01 送给 Server A 之后， PC01 就不理会接下来的工作。而 Server A 接收到这个封包后， 会依据上述的流程，也分析自己的路由信息，然后向后继续传输到正确的目的地主机上头。

Tips: Gateway / Router ：网关/路由器的功能就是在负责不同网域之间的封包转递 (IP Forwarding)，由于路由器具有 IP Forwarding 的功能，并且具有管理路由的能力， 所以可以将来自不同网域之间的封包进行转递的功能。此外，你的主机与你主机设定的 Gateway 必定是在同一个网段内喔！



大致的情况就是这样，所以每一部主机里面都会存在着一个路由表 (Route table)，数据的传递将依据这个路由表进行传送！而一旦封包已经经由路由表的规则传送出去后， 那么主机本身就已经不再管封包的流向了，因为该封包的流向将是下一个主机 (也就是那部 Router) 来进行传送，而 Router 在传送时，也是依据 Router 自己的路由表来判断该封包应该经由哪里传送出去的！整体来说，数据传送有点像这样：

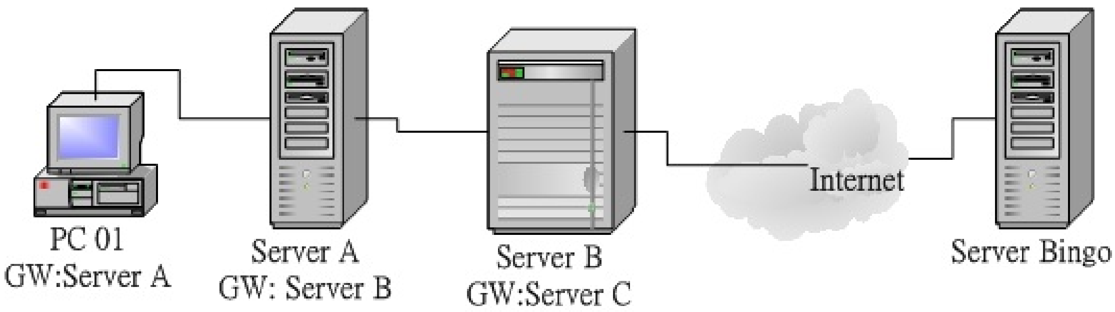


图 2.3-3、路由的概念

PC 01 要将资料送到 Server Bingo 去，则依据自己的路由表，将该封包送到 Server A 去， Server A 再继续送到 Server B ，然后在一个一个的接力给他送下去，最后总是可以到达 Server Bingo 的。

上面的案例是一个很简单的路由概念，事实上， Internet 上面的路由协议与变化是相当复杂的，因为 Internet 上面的路由并不是静态的，他可以随时因为环境的变化而修订每个封包的传送方向。 举例来说，数年前在新竹因为土木施工导致台湾西部整个网络缆线的中断。 不过南北的网络竟然还是能通，为什么呢？因为路由已经判断出西部缆线的终止， 因此他自动的

导向台湾东部的花莲路线，虽然如此一来绕了一大圈，而且造成网络的大塞车， 不过封包还是能通就是了！这个例子仅是想告诉大家，我们上面提的路由仅是一个很简单的静态路由情况， 如果想要更深入的了解 route ，请自行参考相关书籍喔！ ^_^ 。

此外，在属于 Public 的 Internet 环境中，由于最早时的 IP 分配都已经配置妥当， 所以各单位的路由一经设定妥当后，上层的路由则无须担心啊！IP 的分配可以参考底下的网页：

- 台湾地区 IP 核发情况：http://rms.twnic.net.tw/twnic/User/Member/Search/main7.jsp?Order=inet_aton(Startip)

2.3.6 观察主机路由： route

既然路由是这么的重要，而且『路由一旦设定错误， 将会造成某些封包完全无法正确的送出去！』 所以我们当然需要好好的来观察一下我们主机的路由表啦！还是请再注意一下， 每一部主机都有自己的路由表喔！观察路由表的指令很简单，就是 route ，这个指令挺难的，我们在后面章节再继续的介绍，这里仅说明一些比较简单的用法：

```
[root@www ~]# route [-n]
选项与参数：
-n ： 将主机名以 IP 的方式显示

[root@www ~]# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.0     *               255.255.255.0   U     0      0        0 eth0
127.0.0.0       *               255.0.0.0       U     0      0        0 lo
default         192.168.0.254   0.0.0.0         UG    0      0        0 eth0

[root@www ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
127.0.0.0       0.0.0.0         255.0.0.0       U     0      0        0 lo
0.0.0.0         192.168.0.254   0.0.0.0         UG    0      0        0 eth0

# 上面输出的数据共有八个字段，你需要注意的有几个地方：
# Destination ：其实就是 Network 的意思；
# Gateway     ：就是该接口的 Gateway 那个 IP 啦！若为 0.0.0.0 表示不需要额外的 IP；
# Genmask     ：就是 Netmask 啦！与 Destination 组合成为一部主机或网域；
# Flags       ：共有多个旗标可以来表示该网域或主机代表的意义：
#               U：代表该路由可用；
#               G：代表该网域需要经由 Gateway 来帮忙转递；
#               H：代表该行路由为一部主机，而非一整个网域；
# Iface       ：就是 Interface (接口) 的意思。
```

在上面的例子当中，鸟哥是以 PC 01 这部主机的路由状态来进行说明。由于 PC 01 为 192.168.0.0/24 这个网域，所以主机已经建立了这个网域的路由了，那就是『 192.168.0.0 *255.255.255.0 ...* 』那一行所显示的讯息！当你下达 *route* 时， 屏幕上说明了这部机器上面共有三个路由规则，第一栏为『目的地的网域』，例如 *192.168.0.0* 就是一个网域咯，最后一栏显示的是『要去到这个目的地要使用哪一个网络接口！』例如 *eth0* 就是网络卡的装置代号

啦。如果我们要传送的封包在路由规则里面的 *192.168.0.0/255.255.255.0* 或者 *127.0.0.0/255.0.0.0* 里面时，因为第二栏 *Gateway* 为 ，所以就会直接以后面的网络接口来传送出去，而不透过 Gateway 咯！

万一我们要传送的封包目的地 IP 不在路由规则里面，那么就会将封包传送到『default』所在的那个路由规则去，也就是 192.168.0.254 那个 Gateway 喔！所以，几乎每一部主机都会有一个 default gateway 来帮他们负责所有非网域内的封包转递！这是很重要的概念喔！^_^！关于更多的路由功能与设定方法，我们在第八章当中会再次的提及呢！

2.3.7 IP 与 MAC：链结层的 ARP 与 RARP 协定

现在我们知道 Internet 上面最重要的就是那个 IP 了，也会计算所谓的局域网络与路由。 但是，事实上用在传递数据的明明就是以太网络啊！以太网络主要是用网卡卡号 (MAC) 的嘛！这就有问题啦！那这两者 (IP 与 MAC) 势必有一个关连性存在吧？没错！那就是我们要谈到的 ARP (Address Resolution Protocol, 网络地址解析) 协议，以及 RARP (Revers ARP, 反向网络地址解析)

当我们想要了解某个 IP 其实是设定于某张以太网络卡上头时，我们的主机会对整个区网发送出 ARP 封包， 对方收到 ARP 封包后就会回传他的 MAC 给我们，我们的主机就会知道对方所在的网卡，那接下来就能够开始传递数据啰。 如果每次要传送都得要重新来一遍这个 ARP 协定那不是很烦？因此，当使用 ARP 协议取得目标 IP 与他网卡卡号后， 就会将该笔记录写入我们主机的 ARP table 中 (内存内的数据) 记录 20 分钟 (注14)。

例题：如何取得自己本机的网卡卡号 (MAC)答：

```
在 Linux 环境下
[root@www ~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:01:03:43:E5:34
          inet addr:192.168.1.100   Bcast:192.168.1.255   Mask:255.255.255.0
          inet6 addr: fe80::201:3ff:fe43:e534/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
.....

在 Windows 环境下
C:\Documents and Settings\admin..&gt; ipconfig /all
....
        Physical Address. . . . . . . . . : 00-01-03-43-E5-34
....
```

那如何取得本机的 ARP 表格内的 IP/MAC 对应数据呢？就透过 arp 这个指令吧！

```
[root@www ~]# arp -[nd] hostname
[root@www ~]# arp -s hostname(IP) Hardware_address
选项与参数：
-n ：将主机名以 IP 的型态显示
-d ：将 hostname 的 hardware_address 由 ARP table 当中删除掉
-s ：设定某个 IP 或 hostname 的 MAC 到 ARP table 当中

范例一：列出目前主机上面记载的 IP/MAC 对应的 ARP 表格
[root@www ~]# arp -n
Address            HWtype  HWaddress           Flags Mask    Iface
192.168.1.100      ether   00:01:03:01:02:03     C           eth0
192.168.1.240      ether   00:01:03:01:DE:0A     C           eth0
192.168.1.254      ether   00:01:03:55:74:AB     C           eth0

范例二：将 192.168.1.100 那部主机的网卡卡号直接写入 ARP 表格中
[root@www ~]# arp -s 192.168.1.100  01:00:2D:23:A1:0E
# 这个指令的目的在建立静态 ARP
```

如同上面提到的，当你发送 ARP 封包取得的 IP/MAC 对应，这个记录的 ARP table 是动态的信息 (一般保留 20 分钟)，他会随时随着你的网域里面计算机的 IP 更动而变化，所以，即使你常常更动你的计算机 IP，不要担心，因为 ARP table 会自动的重新对应 IP 与 MAC 的表格内容！但如果你有特殊需求的话， 也可以利用『 arp -s 』这个选项来定义静态的 ARP 对应喔！

2.3.8 ICMP 协定

ICMP 的全名是『 Internet Control Message Protocol, 因特网讯息控制协议 』。 基本上，ICMP 是一个错误侦测与回报的机制，最大的功能就是可以确保我们网络的联机状态与联机的正确性！ ICMP 也是网络层的重要封包之一，不过，这个封包并非独立存在，而是纳入到 IP 的封包中！也就是说， ICMP 同样是透过 IP 封包来进行数据传送的啦！因为在 Internet 上面有传输能力的就是 IP 封包啊！ ICMP 有相当多的类别可以侦测与回报，底下是比较常见的几个 ICMP 的类别 (Type)：

类别代号	类别名称与意义
0	Echo Reply (代表一个响应信息)
3	Destination Unreachable (表示目的地不可到达)
4	Source Quench (当 router 的负载过高时，此类别码可用来让发送端停止发送讯息)
5	Redirect (用来重新导向路由路径的信息)
8	Echo Request (请求响应消息)
11	Time Exceeded for a Datagram (当数据封包在某些路由传送的现象中造成逾时状态，此类别码可告知来源该封包已被忽略的讯息)
12	Parameter Problem on a Datagram (当一个 ICMP 封包重复之前的错误时，会回复来源主机关于参数错误的讯息)
13	Timestamp Request (要求对方送出时间讯息，用以计算路由时间的差异，以满足同步性协议的要求)
14	Timestamp Reply (此讯息纯粹是响应 Timestamp Request 用的)
15	Information Request (在 RARP 协议应用之前，此讯息是用来在开机时取得网络信息)
16	Information Reply (用以响应 Infromation Request 讯息)
17	Address Mask Request (这讯息是用来查询子网 mask 设定信息)
18	Address Mask Reply (响应子网 mask 查询讯息的)

那么我们是如何利用 ICMP 来检验网络的状态呢？最简单的指令就是 ping 与 traceroute 了，这两个指令可以透过 ICMP 封包的辅助来确认与回报网络主机的状态。在设定防火墙的时候， 我们最容易忽略的就是这个 ICMP 的封包了，因为只会记住 TCP/UDP 而已～事实上，ICMP 封包可以帮助联机的状态回报，除了上述的 8 可以考虑关闭之外，基本上，ICMP 封包也不应该全部都挡掉喔！

2.4 TCP/IP 的传输层相关封包与数据

网络层的 IP 封包只负责将数据送到正确的目标主机去，但这个封包到底会不会被接受，或者是有没有被正确的接收， 那就不是 IP 的任务啦！那是传送层的任务之一。从 图 2.1-4 我们可以看到传送层有两个重点， 一个是连接导向的 TCP 封包，一个是非连接导向的 UDP 封包，这两个封包很重要啊！数据能不能正确的被送达目的， 与这两个封包有关喔！

2.4.1 可靠联机的 TCP 协议

在前面的 OSI 七层协议当中，在网络层的 IP 之上则是传送层，而传送层的数据打包成什么？最常见的就是 TCP 封包了。这个 TCP 封包数据必须要能够放到 IP 的数据袋当中才行喔！ 所以，我们将图 2.1-4 简化一下，将 MAC, IP 与 TCP 的封包数据这样看：



图 2.4-1、各封包之间的相关性

想当然尔，TCP 也有表头数据来记录该封包的相关信息啰？没错啦～ TCP 封包的表头是长这个样子的：

4 bits	6 bits	6 bits	8 bits	8 bits
Source Port	Destination Port			
Sequence Number				
Acknowledge Number				
Data Offset	Reserved	Code	Window	
Checksum	Urgent Pointer			
Options	Padding			
Data				

图 2.4-2、TCP 封包的表头资料

上图就是一个 TCP 封包的表头数据，各个项目以 Source Port, Destination Port 及 Code 算是比较重要的项目，底下我们就分别来谈一谈各个表头数据的内容吧！

- Source Port & Destination Port (来源埠口 & 目标端口) 什么是埠口(port)？我们知道 IP 封包的传送主要是藉由 IP 地址连接两端， 但是到底这个联机的通道是连接到哪里去呢？没错！就是连接到 port 上头啦！ 举例来说，鸟哥的网站有开放 WWW 服务器，这表示鸟站的主机必须要启动一个可以让 client 端连接的端口，这个端口就是 port (中文翻译成为埠口)。同样的，客户端想要连接到鸟哥的鸟站时，就必须要在 client 主机上面启动一个 port ，这样这两个主机才能够利用这条『通道』来传递封包数据喔！这个目标与来源 port 的纪录，可以说是 TCP 封包上最重要的参数了！
- Sequence Number (封包序号) 由于 TCP 封包必须要带入 IP 封包当中，所以如果 TCP 数据太大时(大于 IP 封包的容许程度)， 就得要进行分段。这个 Sequence Number 就是记录每个封包的序号，可以让收受端重新将 TCP 的数据组合起来。
- Acknowledge Number (回应序号) 为了确认主机端确实有收到我们 client 端所送出的封包数据，我们 client 端当然希望能够收到主机方面的响应，那就是这个 Acknowledge Number 的用途了。 当 client 端收到这个确认码时，就能够确定之前传递的封包已经被正确的收下了。
- Data Offset (资料补偿) 在图 2.4-2 倒数第二行有个 Options 字段对吧！那个 Options 的字段长度是非固定的，而为了要确认整个 TCP 封包的大小，就需要这个标志来说明整个封包区段的起始位置。
- Reserved (保留) 未使用的保留字段。
- Code (Control Flag, 控制标志码) 当我们在进行网络联机的时候，必须要说明这个联机的状态，好让接收端了解这个封包的主要动作。 这可是一个非常重要的句柄喔！这个字段共有 6 个 bits ，分别代表 6 个句柄，若为 1 则为启动。分别说明如下：
    - URG(Urgent)：若为 1 则代表该封包为紧急封包， 接收端应该要紧急处理，且图 2.4-1 当中的 Urgent Pointer 字段也会被启用。
    - ACK(Acknowledge)：若为 1 代表这个封包为响应封包， 则与上面提到的 Acknowledge Number 有关。
    - PSH(Push function)：若为 1 时，代表要求对方立即传送缓冲区内的其他对应封包，而无须等待缓冲区满了才送。
    - RST(Reset)：如果 RST 为 1 的时候，表示联机会被马上结束，而无需等待终止确认手续。这也就是说， 这是个强制结束的联机，且发送端已断线。
    - SYN(Synchronous)：若为 1，表示发送端希望双方建立同步处理， 也就是要求建立联机。通常带有 SYN 标志的封包表示『主动』要连接到对方的意思。
    - FIN(Finish)：若为 1 ，表示传送结束，所以通知对方数据传毕， 是否同意断线，只是发送者还在等待对方的响应而已。

其实每个项目都很重要，不过我们这里仅对 ACK/SYN 有兴趣而已，这样未来在谈到防火墙的时候，你才会比较清楚为啥每个 TCP 封包都有所谓的『状态』条件！那就是因为联机方向的不同所致啊！底下我们会进一步讨论喔！ 至于其他的数据，就得请您自行查询网络相关书籍了！

- Window (滑动窗口) 主要是用来控制封包的流量的，可以告知对方目前本身有的缓冲器容量(Receive Buffer) 还可以接收封包。当 Window=0 时，代表缓冲器已经额满，所以应该要暂停传输数据。 Window 的单位是 byte。

- Checksum(确认检查码) 当数据要由发送端送出前，会进行一个检验的动作，并将该动作的检验值标注在这个字段上； 而接收者收到这个封包之后，会再次的对封包进行验证，并且比对原发送的 Checksum 值是否相符，如果相符就接受，若不符就会假设该封包已经损毁，进而要求对方重新发送此封包！

- Urgent Pointer(紧急资料) 这个字段是在 Code 字段内的 URG = 1 时才会产生作用。可以告知紧急数据所在的位置。

- Options(任意资料) 目前此字段仅应用于表示接收端可以接收的最大数据区段容量，若此字段不使用， 表示可以使用任意资料区段的大小。这个字段较少使用。

- Padding(补足字段) 如同 IP 封包需要有固定的 32bits 表头一样， Options 由于字段为非固定， 所以也需要 Padding 字段来加以补齐才行。同样也是 32 bits 的整数。

谈完了 TCP 表头数据后，再来让我们了解一下这个表头里面最重要的端口信息吧！

- 通讯端口

在上图的 TCP 表头数据中，最重要的就属那 16 位的两个咚咚，亦即来源与目标的端口。由于是 16 位，因此目标与来源端口最大可达 65535 号 (2 的 16 次方)！ 那这个埠口有什么用途呢？上面稍微提到过，网络是双向的，服务器与客户端要达成联机的话， 两边应该要有一个对应的埠口来达成联机信道，好让数据可以透过这个信道来进行沟通。

那么这个埠口怎么打开呢？就是透过程序的执行！举例来说，鸟哥的网站上，必须要启动一个 WWW 服务器软件， 这个服务器软件会主动的唤起 port 80 来等待客户端的联机。你想要看我网站上的数据，就得要利用浏览器， 填入网址，然后浏览器也会启动一个埠口，并将 TCP 的表头填写目标端口为 80 ，而来源端口是你主机随机启动的一个埠口， 然后将 TCP 封包封装到 IP 后，送出到网络上。等鸟站主机接收到你这个封包后，再依据你的埠口给予回应。

这么说你或许不好理解，我们换个说法好了。假如 IP 是网络世界的门牌，那么这个埠口就是那个门牌号码上建筑物的楼层！ 每个建筑物都有 1~65535 层楼，你需要什么网络服务，就得要去该对应的楼层取得正确的资料。但那个楼层里面有没有人在服务你呢？ 这就得要看有没有程序真的在执行啦。所以，IP 是门牌，TCP 是楼层，真正提供服务的， 是在该楼层的那个人 (程序)！

Tips: 曾经有一个朋友问过我说：『一部主机上面这么多服务，那我们跟这部主机进行联机时，该主机怎么知道我们要的数据是 WWW 还是 FTP 啊？』就是透过埠口啊！因为每种 Client 软件他们所需要的数据都不相同，例如上面提到的浏览器所需要的数据是 WWW ，所以该软件默认就会向服务器的 port 80 索求数据；而如果你是使用 filezilla 来进行与服务器的 FTP 数据索求时， filezilla 当然预设就是向服务器的 FTP 相关埠口 (预设就是 port 21) 进行连接的动作啦！所以当然就可以正确无误的取得 Client 端所需要的数据了

再举个例子来说，一部主机就好像是一间多功能银行，该银行内的每个负责不同业务的窗口就好像是通讯端口， 而我们民众就好像是 Client 端来的封包。当你进入银行想要缴纳信用卡账单时， 一到门口服务人员就会指示你直接到该窗口去缴纳，当然，如果你是要领钱，服务人员就会请你到领钱的窗口去填写数据， 你是不会跑错的对吧！ ^_^。万一跑错了怎么办？呵呵！当然该窗口就会告诉你『我不负责这个业务，你请回去！』， 呵呵！所以该次的联机就会『无法成功』咯！



- 特权埠口 (Privileged Ports)

你现在了解了埠口的意义后，再来想想，网络既然是双向的，一定有一个发起端。问题是，到底要联机到服务器取得啥玩意儿？ 也就是说，哪支程序应该在哪个端口执行，以让大家都知道该埠口就是提供哪个服务，如此一来，才不会造成广大用户的困扰嘛！ 所以啰，Internet 上面已经有很多规范好的固定 port (well-known port)， 这些 port number 通常小于 1024 ，且是提供给许多知名的网络服务软件用的。 在我们的 Linux 环境下，各网络服务与 port number 的对应默认给他写在 /etc/services 档案内喔！ 底下鸟哥列出几个常见的 port number 与网络服务的对应：

端口	服务名称与内容
20	FTP-data，文件传输协议所使用的主动数据传输端口
21	FTP，文件传输协议的命令通道
22	SSH，较为安全的远程联机服务器
23	Telnet，早期的远程联机服务器软件
25	SMTP，简单邮件传递协议，用在作为 mail server 的埠口
53	DNS，用在作为名称解析的领域名服务器
80	WWW，这个重要吧！就是全球信息网服务器
110	POP3，邮件收信协议，办公室用的收信软件都是透过他
443	https，有安全加密机制的WWW服务器

另外一点比较值得注意的是，小于 1024 以下的埠口要启动时， 启动者的身份必须要是 root 才行，所以才叫做特权埠口嘛！这个限制挺重要的，大家不要忘记了喔！ 不过如果是 client 端的话，由于 client 端都是主动向 server 端要数据， 所以 client 端的 port number 就使用随机取一个大于 1024 以上且没有在用的 port number。

- Socket Pair

由于网络是双向的，要达成联机的话得要服务器与客户端均提供了 IP 与埠口才行。 因此，我们常常将这个成对的数据称之为 Socket Pair 了！

- 来源 IP + 来源埠口 (Source Address + Source Port)
- 目的 IP + 目的埠口 (Destination Address + Destination Port)

由于 IP 与埠口常常连在一起说明，因此网络寻址常常使用『 IP:port 』来说明，例如想要连上鸟哥的网站时， 正确的鸟哥网站写法应该是：『 linux.vbird.org:80 』才对！

2.4.2 TCP 的三向交握

TCP 被称为可靠的联机封包，主要是透过许多机制来达成的，其中最重要的就是三向交握的功能。 当然， TCP 传送数据的机制非常复杂，有兴趣的朋友请自行参考相关网络书籍。 OK，那么如何藉由 TCP 的表头来确认这个封包有实际被对方接收，并进一步与对方主机达成联机？ 我们以底下的图示来作为说明。

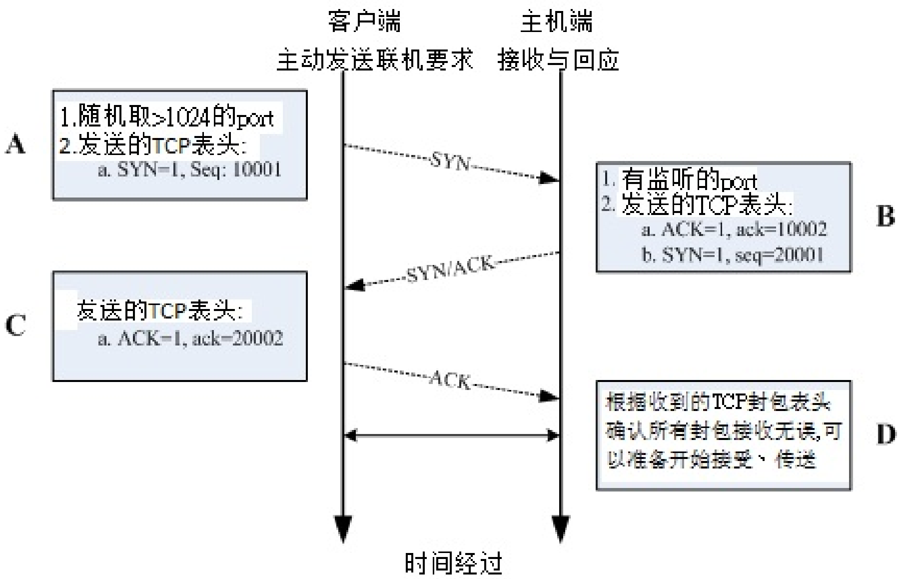


图 2.4-3、三向交握之封包连接模式

在上面的封包连接模式当中，在建立联机之前都必须要通过三个确认的动作， 所以这种联机方式也就被称为三向交握(Three-way handshake)。 那么我们将整个流程依据上面的 A, B, C, D 四个阶段来说明一下：

- A:封包发起 当客户端想要对服务器端联机时，就必须要送出一个要求联机的封包，此时客户端必须随机取用一个大于 1024 以上的端口来做为程序沟通的接口。然后在 TCP 的表头当中，必须要带有 SYN 的主动联机(SYN=1)，并且记下发送出联机封包给服务器端的序号 (Sequence number = 10001) 。
- B:封包接收与确认封包传送 当服务器接到这个封包，并且确定要接收这个封包后，就会开始制作一个同时带有 SYN=1, ACK=1 的封包， 其中那个 acknowledge 的号码是要给 client 端确认用的，所以该数字会比(A 步骤)里面的 Sequence 号码多一号 (ack = 10001+1 = 10002)， 那我们服务器也必须要确认客户端确实可以接收我们的封包才行，所以也会发送出一个 Sequence (seq=20001) 给客户端，并且开始等待客户端给我们服务器端的回应喔！
- C:回送确认封包 当客户端收到来自服务器端的 ACK 数字后 (10002) 就能够确认之前那个要求封包被正确的收受了， 接下来如果客户端也同意与服务器端建立联机时，就会再次的发送一个确认封包 (ACK=1) 给服务器，亦即是 acknowledge = 20001+1 = 20002 啰。
- D:取得最后确认 若一切都顺利，在服务器端收到带有 ACK=1 且 ack=20002 序号的封包后，就能够建立起这次的联机了。

也就是说，你必须要了解『网络是双向的』这个事实！ 所以不论是服务器端还是客户端，都必须要透过一次 SYN 与 ACK 来建立联机，所以总共会进行三次的交谈！ 在设定防火墙或者是追踪网络联机的问题时，这个『双向』的概念最容易被忽略， 而常常导致无法联机成功的问题啊！切记切记！

Tips: 鸟哥上课谈到 TCP 最常做的事就是，叫一个同学起来，实际表演三向交握给大家看！ 1. 鸟哥说：A同学你在不在？ 2. A同学说：我在！那鸟哥你在不在？ 3. 鸟哥说：我也在 此时两个人就确认彼此都可以听到对方在讲啥，这就是可靠联机啦！ ^_^



2.4.3 非连接导向的 UDP 协议

UDP 的全名是：『User Datagram Protocol, 用户数据流协议』，UDP 与 TCP 不一样， UDP 不提供可靠的传输模式，因为他不是面向连接的一个机制，这是因为在 UDP 的传送过程中，接受端在接受到封包之后，不会回复响应封包 (ACK) 给发送端，所以封包并没有像 TCP 封包有较为严密的检查机制。至于 UDP 的表头资料如下表所示：

16 bits	16 bits
Source Port	Destination Port
Message Length	Checksum
Data	

图 2.4-4、UDP 封包的表头资料

TCP 封包确实是比较可靠的，因为通过三向交握嘛！不过，也由于三向交握的缘故， TCP 封包的传输速度会较慢。 至于 UDP 封包由于不需要确认对方是否有正确的收到数据，故表头数据较少，所以 UDP 就可以在 Data 处填入更多的资料了。同时 UDP 比较适合需要实时反应的一些数据流，例如影像实时传送软件等， 就可以使用这类的封包传送。也就是说， UDP 传输协议并不考虑联机要求、联机终止与流量控制等特性， 所以使用的时机是当数据的正确性不很重要的情况，例如网络摄影机！

另外，很多的软件其实是同时提供 TCP 与 UDP 的传输协议的，举例来说，查询主机名的 DNS 服务就同时提供了 UDP/TCP 协议。由于 UDP 较为快速，所以我们 client 端可以先使用 UDP 来与服务器联机。 但是当使用 UDP 联机却还是无法取得正确的数据时，便转换为较为可靠的 TCP 传输协议来进行数据的传输啰。 这样可以同时兼顾快速与可靠的传输说！

Tips: 那么上课时怎么介绍 UDP 呢？很简单喔！鸟哥就会说：『现在老师就是在进行 UDP 的传送，因为老师一直讲一直讲， 俺也没有注意到你有没有听到，也不需要等待你的响应封包！就这样一直讲！当然，你没有听到鸟哥讲啥，我也不会知道...』



2.4.4 网络防火墙与 OSI 七层协定

由上面的说明当中，我们知道数据的传送其实就是封包的发出与接受的动作啦！并且不同的封包上面都有不一样的表头 (header)，此外，封包上面通常都会具有四个基本的信息，那就是 socket pair 里面提到的『来源与目的 IP 以及来源与目的端的 port number』 。当然啦，如果是可靠性联机的 TCP 封包，还包含 Control Flag 里面的 SYN/ACK 等等重要的信息呢！好了，开始动一动脑筋，有没有想到『网络防火墙』的字眼啊？

封包过滤式的网络防火墙可以抵挡掉一些可能有问题的封包， Linux 系统上面是怎么挡掉封包的呢？其实说来也是很简单，既然封包的表头上面已经有这么多的重要信息， 那么我就利用一些防火墙机制与软件来进行封包表头的分析，并且设定分析的规则，当发现某些特定的 IP 、特定的埠口或者是特定的封包信息(SYN/ACK等等)，那么就将该封包给他丢弃， 那就是最基本的防火墙原理了！

举例来说，大家都知道 Telnet 这个服务器是挺危险的，而 Telnet 使用的 port number 为 23 ，所以，当我们使用软件去分析要送进我们主机的封包时， 只要发现该封包的目的地是我们主机的 port 23 ，就将该封包丢掉去！那就是最基本的防火墙案例啦！ 如果以 OSI 七层协议来说，每一层可以抵挡的数据有：

- 第二层：可以针对来源与目标的 MAC 进行抵挡；
- 第三层：主要针对来源与目标的 IP ，以及 ICMP 的类别 (type) 进行抵挡；
- 第四层：针对 TCP/UDP 的埠口进行抵挡，也可以针对 TCP 的状态 (code) 来处理。

更多的防火墙信息我们会在第九章防火墙与第七章认识网络安全当中进行更多的说明喔！

2.5 连上 Internet 前的准备事项

讲了这么多，其实我们最需要的仅是『连接上 Internet 』啦！那么在 Internet 上面其实使用的是 TCP/IP 这个通讯协议，所以我们就需要 Public IP 来连接上 Internet 啊！你说对吧～ 不过，你有没有发现一件事，那就是『为啥我不知道 Yahoo 的主机 IP ，但是俺的主机却可以连到 Yahoo 主机上？』 如果你有发现这个问题的话，哈哈！你可以准备开始设定网络啰～ ^_^

2.5.1 用 IP 上网？主机名上网？DNS 系统？

讲完了上头的基本数据，现在你知道要连上 Internet 就得要有 TCP/IP 才行！尤其是那重要的 IP 啊！ 问题是，计算机网络是依据人类的需要来建立的，不过人类对于 IP 这一类的数字并不具有敏感性，即使 IP 已经被简化为十进制了，但是人类就是对数字没有办法啊！怎么办？没关系，反正计算机都有主机名嘛！ 那么我就将主机名与他的 IP 对应起来，未来要连接上该计算机时，只要知道该计算机的主机名就好了，因为 IP 已经对应到主机名了嘛！所以人类也容易记忆文字类的主机名，计算机也可以藉由对应来找到他必须要知道的 IP ，啊！真是皆大欢喜啊！

这个主机名 (Hostname) 对应 IP 的系统，就是鼎鼎有名的 Domain Name System (DNS) 咯！也就是说， DNS 这个服务的最大功能就是在进行『主机名与该主机的 IP 的对应』的一项协议。 DNS 在网络环境当中是相当常被使用到的一项协议喔！举个例子来说，像鸟哥我常常会连到奇摩雅虎的 WWW 网站去看最新的新闻，那么我一定需要将奇摩雅虎的 WWW 网站的 IP 背下来吗？天呐， 鸟哥的忘性这么好，怎么可能将 IP 背下来？！不过，如果是要将奇摩站的主机名背下来的话， 那就容易的多了！不就是 http://tw.yahoo.com 吗？而既然计算机主机只认识 IP 而已， 因此当我在浏览器上面输入了『http://tw.yahoo.com』的时后，我的计算机首先就会藉由向 DNS 主机查询 tw.yahoo.com 的 IP 后，再将查询到的 IP 结果回应给我的浏览器， 那么我的浏览器就可以藉由该 IP 来连接上主机啦！

发现了吗？我的计算机必须要向 DNS 服务器查询 Hostname 对应 IP 的信息 喔！那么那部 DNS 主机的 IP 就必须要在我的计算机里面设定好才行，并且必须要是输入 IP 喔，不然我的计算机怎么连到 DNS 服务器去要求数据呢？呵呵！在 Linux 里面，DNS 主机 IP 的设定就是在 /etc/resolv.conf 这个档案里面啦！

目前各大 ISP 都有提供他们的 DNS 服务器的 IP 给他们的用户，好设定客户自己计算机的 DNS 查询主机， 不过，如果你忘记了或者是你使用的环境中并没有提供 DNS 主机呢？呵呵！没有关系， 那就设定 Hinet 那个最大的 DNS 服务器吧！ IP 是 168.95.1.1 咯！要设定好 DNS 之后，未来上网浏览时，才能使用主机名喔！不然就得一定需要使用 IP 才能上网呢！ DNS 是很重要的，他的原理也顶复杂的，更详细的原理我们在第十九章 DNS 服务器里面进行更多更详细的说明喔！这里仅提个大纲！

2.5.2 一组可以连上 Internet 的必要网络参数

从上面的所有说明当中，我们知道一部主机要能够使用网络，必须要有 IP ，而 IP 的设定当中，就必须要有 IP, Network, Broadcast, Netmask 等参数，此外，还需要考虑到路由里面的 Default Gateway 才能够正确的将非同网域的封包给他传送出去。 另外，考虑到主机名与 IP 的对应，所以你还必须要给予系统一个 DNS 服务器的 IP 才行～ 所以说，一组合理的网络设定需要哪些数据呢？呵呵！就是：

- IP
- Netmask
- Network
- Broadcast
- Gateway
- DNS

其中，由于 Network 与 Broadcast 可以经由 IP/Netmask 的计算而得到，因此需要设定于你 PC 端的网络参数， 主要就是 IP, Netmask, Default Gateway, DNS 这四个就是了！

没错！就是这些数据！如果你是使用 ADSL 拨接来上网的话，上面这些数据都是由 ISP 直接给你的，那你只要使用拨接程序进行拨接到 ISP 的工作之后， 这些数据就自动的在你的主机上面设定完成了！但是如果是固定制 (如学术网络) 的话，那么就得自行使用上面的参数来设定你的主机啰！缺一不可呢！以 192.168.1.0/24 这个 Class C 为例的话，那么你就必须要在你的主机上面设定好底下的参数：

- IP: 由 192.168.1.1~192.168.1.254
- Netmask: 255.255.255.0
- Network: 192.168.1.0
- Broadcast: 192.168.1.255
- Gateway: 每个环境都不同，请自行询问网络管理员
- DNS: 也可以直接设定成 168.95.1.1

2.6 重点回顾：

- 虽然目前的网络媒体多以以太网络为标准，但网络媒体不只有以太网络而已；
- Internet 主要是由 Internet Network Information Center (INTERNIC) 所维护；
- 以太网络的 RJ-45 网络线，由于 568A/568B 接头的不同而又分为并行线与跳线；
- 以太网络上最重要的传输数据为 Carrier Sense Multiple Access with Collision Detect (CSMA/CD) 技术， 至于传输过程当中，最重要的 MAC 讯框内以硬件地址 (hardware address) 数据最为重要；
- 透过八蕊的网络线 (Cat 5 以上等级)，现在的以太网络可以支持全双工模式；
- OSI 七层协议为一个网络模型 (model) ，并非硬性规定。这七层协议可以协助软硬件开发有一个基本的准则， 且每一分层各自独立，方便使用者开发；
- 现今的网络基础是架构在 TCP/IP 这个通讯协议上面；
- 数据链结层里重要的信息为 MAC (Media Access Control)，亦可称为硬件地址，而 ARP Table 可以用来对应 MAC 与软件地址 (IP) ；
- 在网络媒体方面， Hub 为共享媒体，因此可能会有封包碰撞的问题，至于 Switch 由于加入了 switch port 与 MAC 的对应，因此已经克服了封包碰撞的问题，也就是说，Switch 并不是共享媒体；
- IP 为 32 bits 所组成的，为了适应人类的记忆，因此转成四组十进制的数据；
- IP 主要分为 Net ID 与 Host ID 两部份，加上 Netmask 这个参数后，可以设定『网域』的概念；
- 根据 IP 网域的大小，可将 IP 的等级分为 A, B, C 三种常见的等级；
- Loopback 这个网段在 127.0.0.0/8 ，用在每个操作系统内部的循环测试中。
- 网域可继续分成更小的网域 (subnetwork)，主要是透过将 Host_ID 借位成为 Net_ID 的技术；
- IP 只有两种，就是 Public IP 与 Private IP ，中文应该翻译为 公共 IP 与 私有(或保留) IP，私有 IP 与私有路由不可以直接连接到 Internet 上；
- 每一部主机都有自己的路由表，这个路由表规定了封包的传送途径，在路由表当中，最重要者为默认的通讯闸 (Gateway/Router)；
- TCP 协议的表头数据当中，那个 Code (control flags) 所带有的 ACK, SYN, FIN 等为常见的旗标， 可以控制封包的联机成功与否；
- TCP 与 IP 的 IP address/Port 可以组成一对 socket pair
- 网络联机都是双向的，在 TCP 的联机当中，需要进行客户端与服务器端两次的 SYN/ACK 封包发送与确认， 所以一次 TCP 联机确认时，需要进行三向交握的流程；
- UDP 通讯协议由于不需要联机确认，因此适用于快速实时传输且不需要数据可靠的软件中，例如实时通讯；
- ICMP 封包最主要的功能在回报网络的侦测状况，故不要使用防火墙将他完全挡掉；
- 一般来说，一部主机里面的网络参数应该具备有：IP, Netmask, Network, Broadcast, Gateway, DNS 等；
- 在主机的 port 当中，只有 root 可以启用小于 1024 以下的 port ；

- DNS 主要的目的在于进行 Hostname 对应 IP 的功能；

2.7 本章习题

- 在 ISP 提供的网络服务中，他们提到传输速度为 1.5M/382K ，请问这个数据的单位为何？数据单位为 bits/second, 与惯用的 bytes 差 8 倍。
- 什么是 MAC (Media Access Control) ，MAC 主要的功能是什么？Media Access Control 的缩写，为以太网络硬件讯框的规格，以太网络就是以 MAC 讯框进行数据的传送。 目前 MAC 也常被用为以太网络卡卡号的代称。
- 什么是封包碰撞？为什么会发生封包碰撞？当主机要使用网络时，必须要先进行 CSMA/CD 监听网络，如果(1)网络使用频繁 (2)网络间隔太大， 则可能会发生监听时均显示无主机使用，但发出封包后却发生同步发送封包的情况，此时两个封包就会产生碰撞， 造成数据损毁。
- ARP Table 的作用为何？如何在我的 Linux 察看我的 ARP 表格？ARP 协议主要在分析 MAC 与 IP 的对应，而解析完毕后的数据会存在系统的内存中， 下次要传送到相同的 IP 时，就会主动的直接以该 MAC 传送，而不发送广播封包询问整个网域了。 利用 arp -n 即可
- 简略说明 Netmask 的作用与优点；Netmask 可以用来区分网域，且 Netmask 可以有效的增加网络的效率，这是因为 Netmask 可以定义出一个网域的大小，那么 broadcast 的时间就可以降低很多！一般来说， 我们如果要将一个大网域再细分为小网域，也需要藉由 Netmask 来进行 subnet 的切割。
- 我有一组网域为： 192.168.0.0/28 ，请问这个网域的 Network, Netmask, Broadcast 各为多少？而可以使用的 IP 数量与范围各是多少？因为共有 28 个 bits 是不可动的，所以 Netmask 地址的最后一个数字为 11110000，也就是 (128+64+32+16=240) ，所以： Network：192.168.0.0 Netmask：255.255.255.240 Broadcast：192.168.0.15 IP：由 192.168.0.1 ~ 192.168.0.14 共 14 个可用 IP 喔！
- 承上题，如果网域是 192.168.0.128/29 呢？因为是 29 个 bits 不可动，所以最后一个 Netmask 的地址为： 11111000 也就是 (128+64+32+16+8=248)，所以： Network：192.168.0.128 Netmask：255.255.255.248 Broadcast：192.168.0.135 IP：由 192.168.0.129 ~ 192.168.0.134 共 6 个可用的 IP 喔！

- 我要将 192.168.100.0/24 这个 Class C 的网域分为 4 个子域，请问这四个子域要如何表示？既然要分为四个网域，也就是还需要藉助 Netmask 的两个 bits (2的2次方为4 啊！)，所以 Netmask 会变成 255.255.255.192 ，每个子域会有 256/4=64 个 IP ，而必须要扣除 Network 与 Broadcast ，所以每个子域会有 62 个可用 IP 喔！因此，四个子域的表示方法为： 192.168.100.0/26, 192.168.100.64/26, 192.168.100.128/26, 192.168.100.192/26。

- 如何观察 Linux 主机上面的路由信息 (route table)？路由信息的观察可以下达 route 来直接察看！或者是下达 route -n 亦可
- TCP 封包上面的 SYN 与 ACK 标志代表的意义为何？SYN 代表该封包为该系列联机的第一个封包，亦即是主动联机的意思； ACK 则代表该封包为确认封包，亦即是回应封包！

- 什么是三向交握？在哪一种封包格式上面才会有三向交握？使用 TCP 封包才会有三向交握。TCP 封包的三向交握是一个确认封包正确性的重要步骤，通过 SYN, SYN/ACK, ACK 三个封包的确认无误后，才能够建立联机。至于 UDP 封包则没有三向交握喔！

- 试说明何谓有网管？无网管的 switch ？此外，这些 switch 的硬件应算在 OSI 七层协议的第几层？有网管者，会在 switch 内部加入其他的小型 OS，藉以控管 IP 或 MAC 的流通； 通常基础的 switch 仅达控管 MAC ，故为 OSI 第二层(数据链结层)
- 为何 ISP 有时候会谈到『申请固定 8 个 IP ，其中只有 5 个可以用』，你觉得问题出在哪里？ 如果以网域的观念来看，他的 netmask 会是多少？因为如果是一个网域的话，那么八个 IP 前后(Host_ID 全为 0 与 1 的条件)为 Network 及 Broadcast ， 加上一个在 ISP 处的 Gateway ，所以仅有 5 个可以用。因为有 8 个 IP ，所以其 netmask 后八 bits 为 11111000 ，故为 255.255.255.248。
- Internet 协议中共包含 "Network Access Layer", "Internet Layer", "Transport Layer", "Application Layer"， 请将这四层与 OSI 七层协议的内容进行连结 (自行上网查询相关文章说明)；Network Access Layer: 涵盖 Data-Link 及 Physical Layer Internet Layer: 也是 Network Layer Transport Layer: 也是 Transport Layer Application Layer: 涵盖 Application Layer, Persentatin Layer, Session Layer.
- 请自行上网查询关于 NetBIOS 这个通讯协议的相关理论基础，并请说明 NetBIOS 是否可以跨路由？请自行参考网中人的网络基础文章
- 什么是 Socket pair ？包含哪些基本数据？由 IP 封包的 IP address 与 TCP 封包的 port number 达成，分别为目的端的 IP/port 与本地端的 IP/port。
- IP 有一段 A Class 的网段分给系统做为测试用，请问该网段为？设定的名称为？ 127.0.0.0/8, loopback
- ICMP 这个协议最主要的目的为？同时做为『响应』的类别为第几类？做为网络检测之用，为第 8 类 (echo request)
- IP 封包表头有个 TTL 的标志，请问该标志的基本说明为何？其数据有何特性？为该封包的存活时间，该时间每经过一个 node 都会减少一，当 TTL 为 0 时，该封包会被路由器所丢弃。 该数据最大为 255。
- 在 Linux 当中，如何查询每个 port number 对于服务的对应 (filename)/etc/services 档案中有纪录
- 什么是星形联机？优点为何？利用一 hub/switch 链接所有的网络设备的一种联机方式，最大的好处是，每个『网络设备与 switch 之间』都是独立的， 所以所以每个主机故障时均不会影响其他主机的联机。
- 请说明 CSMA/CD 的运作原理？发送流程

  1. 主机欲使用网络时，会先监听网络，若网络没有被使用时，才会准备传送，否则继续监听；
  2. 当数据传送钟，发现有碰撞情况时，则会重新监听网络，并且重新发送一次该封包；
  3. 若重复发生碰撞 16 次，则网络会瘫痪；

  接收流程

1. 主机如果没有在传送数据，则会监听网络，并且主动在接收的状态下；
2. 若接收到一个封包，并且该表头所载 MAC 为本身的网卡卡号，则开始接收该封包，否则将该封包丢弃；
3. 接收过程当中如果发生封包碰撞，则会通知原发送主机碰撞的数据；
4. 封包接收完毕后，会以 MAC 表头所载长度同时分析本封包长度，若发生问题，则会通知对方重新传送。

2.8 参考数据与延伸阅读

特别感谢： 本文在 2002/07 发出之后，收到相当多朋友的关心，也从而发现了自己误会的一些基础的网络理论，真的是感谢好朋友 Netman 兄与 ZMAN 兄的指导！这篇短文在第二版时 (2003/08/03) 做了相当大幅度的修订，与原来的文章 (上次更新日期 2002/09) 已经有一定程度的差异了，第三版又针对整个内容与阅读顺序进行调整 (2010/08)，希望网友们如果有时间的话，能够再次的阅读，以厘清一些基本概念喔！

- 注1：粘添寿着，『Internet 网络原理与实务』，旗标出版社。
- 注2：Robert Breyer & Sean Riley 着，风信子,张民人译，『Switched & Fast 以太网络』，旗标出版社
- 注3：IEEE 标准的网站连结：http://standards.ieee.org/
- 注4：Request For Comment (RFC) 技术文件：http://www.rfc-editor.org/
- 注5：RFC-1122 标准的文件数据：ftp://ftp.rfc-editor.org/in-notes/rfc1122.txt
- 注6：粘添寿老师官网：http://www.tsnien.idv.tw/, 因特网相关课程：http://120.118.165.46/tsnien/network/index.html(强烈建议前往参阅)
- 注7：台湾学术网络简介 (TANET)：http://www.edu.tw/moecc/content.aspx?site_content_sn=1707
- 注8：Study Area 之网络基础：http://www.study-area.org/network/network.htm
- 注9：维基百科对 OSI 协定的说明：http://en.wikipedia.org/wiki/OSI_model
- 注10：Phil Dykstra, Gigabit Ethernet Jumbo Frames：http://sd.wareonearth.com/~phil/jumbo.html
- 注11：Hub 与 Switch 的迷思：http://www.study-area.org/tips/hub_switch.htm
- 注12：管理 IP 的单位与相关说明：http://www.internic.org/, http://www.icann.org/, http://www.iana.org/, http://en.wikipedia.org/wiki/IPv4
- 注13：管理 IP 的单位：http://www.iana.org/, 台湾地区 IP 核发情况：http://rms.twnic.net.tw/twnic/User/Member/Search/main7.jsp?Order=inet_aton(Startip)
- 注14：相关参考数据 『TCP/IP Illustrated, Volume 1 - The Protocols』，W. Richard Stevens ，资策会中文化部门译； http://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing
- PPPoE http://en.wikipedia.org/wiki/Point-to-Point_Protocol_over_Ethernet

---

2002/07/18：第一次完成日期！ 2002/09/26：修改了部分可能引起误解的文章部分！ 2003/08/03：重新编排版面，并且重新检视文章内容，修订文章！ 2003/08/20：增加重点回顾与课后练习 2003/09/06：加入参考用解答 2004/03/16：修订 N-Way 的错误，订正为 Auto MDI/MDIX 的功能！ 2006/02/09：将旧的文章移动到 此处 2006/07/12：参考了粘教授与风信子兄的书籍，修改了很多基础数据喔！还有重点整理，不过，练习尚未更新 2006/07/16：加

入习题练习啰！ 2007/10/21：图14那个 UDP 的表头资料中，16 bits 误植为 16 bytes，感谢讨论区 ricky.liu 的告知！ 2008/04/21：经由网友 chyanlong 兄的指点，IHL 的大小单位误植为 byte，应该是字组 (word) 才对。 2010/07/22：将基于 CentOS4.x 所写的数据放置于此处 2010/08/15：将章节依据 TCP/IP 相关的层级分别介绍，更改的幅度不小喔！ 2011/07/15：将基于 CentOS 5.x 所撰写的文章移动到此处 2013/07/23：修訂重點回顧CSMA單字的錯誤，Sense誤植爲Sence，感谢mubiale告知。 by white

第三章、局域网络架构简介

最近更新日期：2011/07/15

在这一章当中，我们会继续讨论在一个小型企业或家庭里面的小型局域网络规划，以让的所有计算机主机都可以直接利用以太网络进行数据的连接啊！ 一般来说，内部局域网络都希望直接使用私有 IP 来设定沟通环境，直接以简单的星形联机做为网络施工的主要类型， 底下就来谈一谈如何规划你主机在星形联机所应该要放置的状态。最终我们也列出本书所需要的局域网络联机架构图喔！

3.1 局域网络的联机

谈完了第二章网络基础后， 现在就让我们实际的来将家里或者小型企业内部的全部计算机给他连接起来吧！ 当然啦，我们这里主要介绍的是小型局域网络的架构，如果是比较大型的企业内部， 那么将『配线盘、线路设计、墙上网络孔』分别拆开施工的结构化布线会比较妥当。 不过，结构化布线并非本文所想要讨论的，如果你的企业有需求的话，可以向专业人士寻求协助， 举例来说，酷学园(http://phorum.study-area.org)的 ZMAN 兄就是一位很棒的网络布线专家。无论如何，先来将所有的网络硬件联机起来吧！

3.1.1 局域网络的布线规划

从前一章的数据探讨中，你现在应该已经知道局域网络的定义了。大部分狭义的定义中， 都将局域网络定位在一个以星形联机连接的实体网络中，再透过 IP 网段来连接在一起的情况。所以啦，这个联机是怎么连接在一块的，以及 IP 网段是如何规划的，就显的非常重要啰！

记得以前听 ZMAN 大哥某场演讲的时候提到，网络布线是『数十年大计』中最重要的一环，因为『服务器主机能力不够时换主机就好了，Switch 交换力不足时换 switch 就好了， 但如果布线不良，难道要拆掉房子将管线挖出来重新安装设定？』所以说， 最初规划的布线严谨度眞的会影响到未来网络的分布情况啊！

所以说，如果你的企业『整栋大楼需要重新布线』时，眞的非常建议你务必要找寻专业网络布线专家帮忙设计规划， 因为连一个小小的机柜配线箱都有大学问～设计的好的话，每部独立的主机要改线路、 要换插孔都变的很简单！而且主机到墙上插孔的距离也会变的很短，维护也会很方便！线段也会很美观！ 当然啦，如此一来，网络线材的选择也就不能够用太差的！而且网络布线经过折角区时，也需要特别留意施工呐。

但是本文讨论的是一些比较小的局域网络环境，这样的环境可以是在一间办公室内而已， 所以我们这里谈到的大多是比较单纯的布线状态，并没有考虑到办公室外部的环境， 所以参考本文时，请特别留意这种差异性喔！

在这样单纯的环境中，我们可以利用一个以 switch 为中心来串连所有设备的星形联机 (star topology) 架构来设计我们的局域网络啊！在这样的环境中你需要担心的是『那我的 Linux 服务器要放在那个地方？』会考虑 Linux 服务器是因为鸟哥假设你需要在你的局域网络内架设对 Internet 开放网络的服务！ 而 Linux 是否具有 Public IP 对于主机的维护与设定的复杂度有很大的影响，所以当然需要考虑啰！ 底下鸟哥以目前在台湾挺流行的 ADSL 利用电话线路上网的环境来说明几种联机状态。

在底下的环境当中，鸟哥假设我们仅有一条 ADSL 的对外联机，也就是说， 我们的 Linux 与一般 PC (不论何种操作系统) 都是透过同一条线连到 Internet 上面去的。

3.1.1-1 Linux 直接联网-与 PC 同地位

如果你使用的 ADSL 是多 IP 的条件 (例如拨接可以给予 2-8 个 IP 的情况)， 那么最简单的方式就是如下图的联机模式：

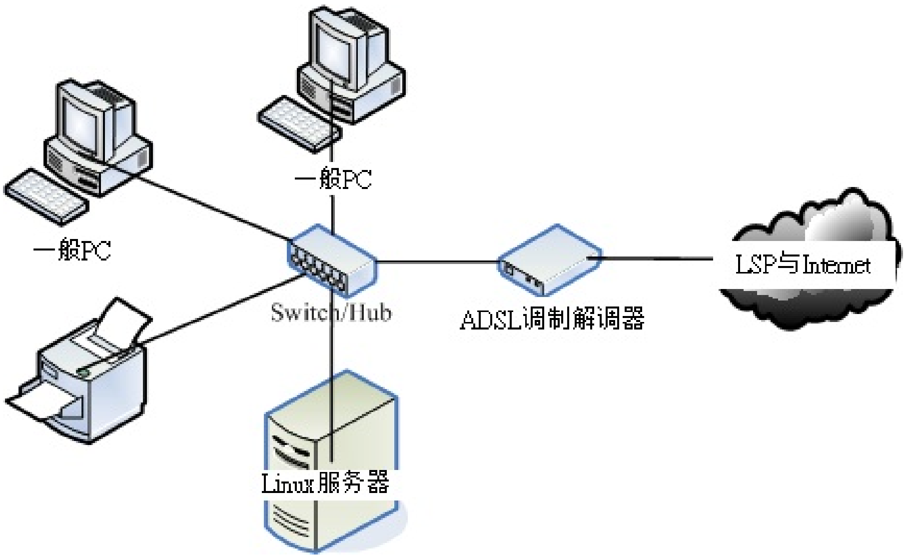


图 3.1-1、Linux 服务器取得 public IP 的联机方式之一(具有多个可用 IP 情况)

在这种联机模式当中， Linux 与一般 PC 或打印机都是同等地位，并没有谁比较『大尾！』^_^ 如果不急着连上 Internet 时，那么每个设备都给予一个同网域的私有 IP 就可以进行网络联机的工作了，你也可以很快乐的使用打印机或者是网络上的芳邻等等工作。 此外， Linux 服务器也可以作为内部的文件服务器或者是打印机服务器等等。

当需要连上 Internet 时，每部计算机 (包括 PC 与 Linux 主机) 都可以自己直接透过拨接连上， 而由于拨接是在每部机器上面『额外增加一个实体的 ppp0 接口』， 此时，你的系统内就会有两个可以使用的 IP 了 (一个是 public 一个是 private IP)。因此， 拨接上网之后每部主机还是可以使用原有的局域网络内的各项服务，而无须更动原本设定妥当的私有 IP 。 这样的情况对于一般家庭使用者来说，可以算是最佳的解决方案啦！因为如果你的 Linux 主机挂点时， 其他个人的 PC 是不会被影响的！

不过这样的环境对于小型企业主来说，却不好管理。因为无法掌握每个员工实际上网的情况， 而且对于防火墙来说，『根本就是一个没有防火墙的环境』，所以， 是没有办法对员工进行任何实际网络的掌控的，并且由于网络内外部 (LAN 与外部环境) 并没有明确的分开，网管人员对于进入客户端的封包是没有任何管理的能力， 所以对于网络安全来说，是很难管控的一种环境啊！因此对于企业来说，不建议这种环境。

3.1.1-2 Linux 直接联网-与一般 PC 分开网域

如果你有多个可用的 public IP ，并且你的 Linux 服务器主要是提供 Internet 的 WWW 或 mail 服务， 而不是作为内部的文件服务器之用，那么将 Linux 服务器与内部的网域分开也是个可行的方法， 而且 Linux 拥有 public IP，在设定与维护上面也不困难，如下所示：

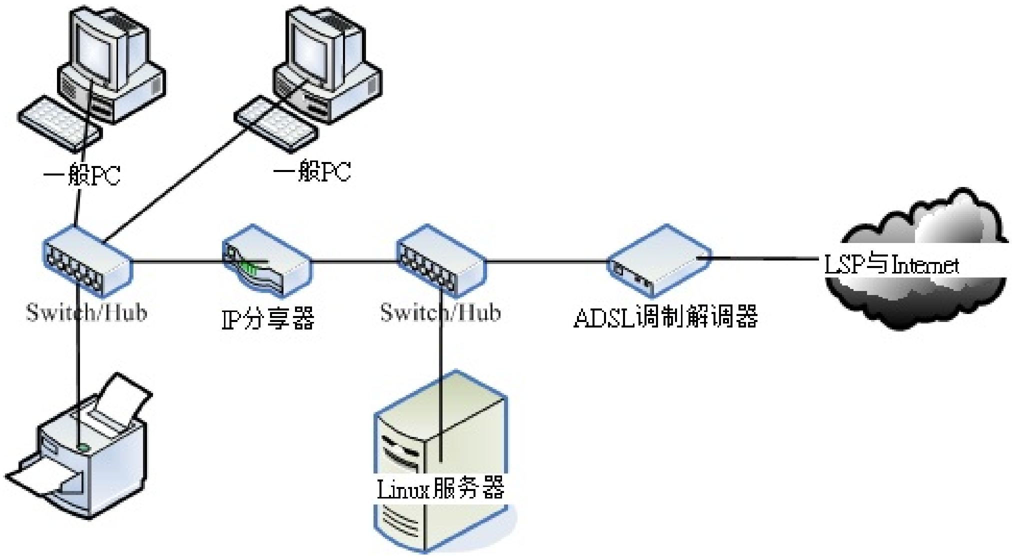


图 3.1-2、Linux 服务器取得 public IP 的联机方式之二(具有多个可用 IP 情况)

所有的 LAN 内的计算机与相关设备都会在同一个网域内，所以在 LAN 内的传输速度是没有问题的， 此外，这些计算机要连出至 Internet 时，必须要透过 IP 分享器，所以你也可以在 IP 分享器上面设定简单的防火墙规则， 如果 IP 分享器可以换更高阶的设备时，那么你就可以在该设备上面架设规则较为完整的防火墙， 对于内部主机有相当程度的管理，并且好维护啊！

3.1.1-3 Linux 直接联网-让 Linux 直接管理 LAN

如果你不想要购买 IP 分享器的话，那么直接利用 Linux 服务器来管理就好了啊！那么你可以这样布线：

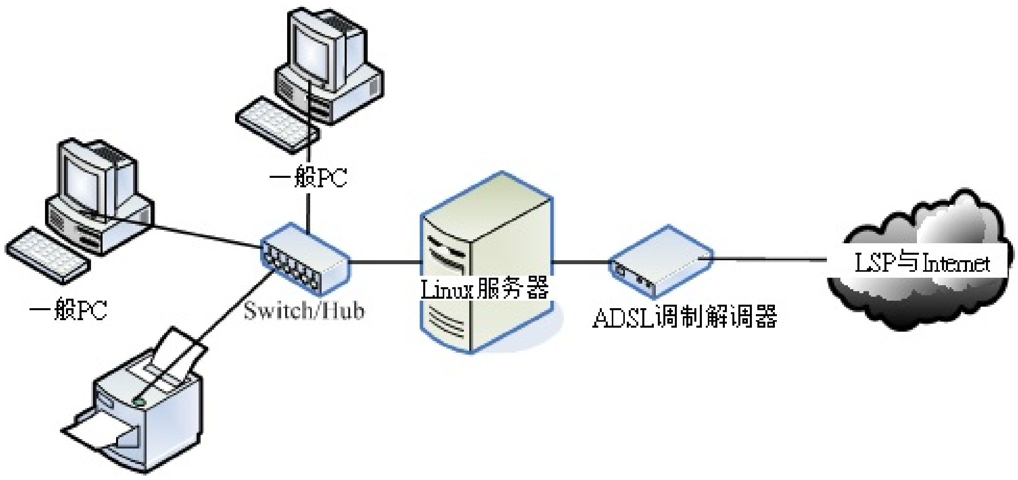


图 3.1-3、让 Linux 管理 LAN 的布线情况

这种情况下，不论你有多少个 IP 都可以适用的，尤其是当你只有一个 public IP 时，就非得使用这种方式不可了。 让 Linux 作为 IP 分享器的功能相当的简单，同时 Linux 必须具备两张网络卡，分别是对外与对内， 由于 Linux 依旧具有 public IP ，所以在服务器的设定与维护上相当的简单，同时 Linux 服务器可以做为内部网域对外的防火墙之用，由于 Linux 防火墙的效能挺不错加上设定也很简单，功能却也是很不错的！ 因此，网络管理人员也较能进行较完善的掌控，并且， Linux 服务器也要比高阶的硬件防火墙便宜多了！ ^_^ 鸟哥个人是比较喜好这种方式的联机啦！

不过，我们都知道『服务器提供的网络服务越单纯越好』， 因为这样一来主机的资源可以完全被某个程序所使用，不会互相影响，而且当主机被攻击时， 也比较能够立即了解是那个环节出了问题。但是如同图 3.1-3 的状况来说的话，由于内部的 LAN 是需要通过 Linux 才能联机出去，所以 Linux 挂点时，整个对外联机就挂了，此外， Linux 的服务可能就太复杂了点，可能会造成维护上的困难度。 但对于小型区网来说，图 3.1-3 这种架构还是可以应付的来的啦！

3.1.1-4 Linux 放在防火墙后-让 Linux 使用 Private IP

我们可以将 Linux 服务器放在 LAN 后面喔！瞎密？我们的 Linux 主机放在 LAN 里面？有没有搞错啊？没搞错啊～ 比较大型的企业通常会将他们的服务器主机放置在机房内，主要是在 LAN 的环境下， 再透过防火墙的封包重新导向的功能，将来自 Internet 的封包先经过防火墙后才进入到服务器， 如此一来可在防火墙端就砍掉一堆莫名其妙的侦测与攻击，当然会比较安全啊！ 这种架构还依防火墙的多寡而又可分为非军事区 (DMZ) 的配置，不过，太麻烦了～不建议初学者直接使用。 底下我们仅介绍较简单的架构来说明：

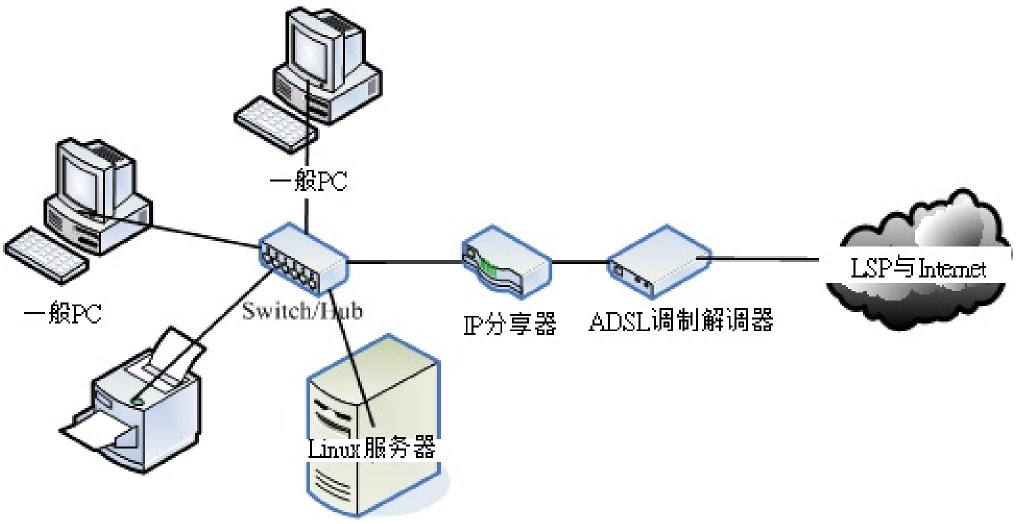


图 3.1-4、Linux 主机放在 LAN 里面的布线情况

这里我们以一个较简单的图示来说明，所以利用的还是 IP 分享器，可能的话， 你可以将 IP 分享器换成 Linux 主机来架设防火墙，也是一个不错的选择啊！ 反正现在计算机天天在升级，升级后的旧配备其实就可以作为 Linux 防火墙之用了！ 反正防火墙又不需要什么硬盘与

强效的显示或者 CPU， 只要有不错的网络接口就能够达到不错的防火墙效能了。

不过这里得再次的强调， Linux 服务器主机若放在 LAN 里面 (使用 private IP)，则当你要对 Internet 提供网络服务时， 防火墙的规则将变的相当复杂，因为需要进行封包转递的任务，在某些比较麻烦的协议当中， 可能会造成设定方面的困扰。所以，在你初接触 Linux 服务器时， 不建议新手使用这种联机架构，避免由于失去信心而没有动力学习～(@_@)。

每种联机的方式都有其适用的使用者群，所以没有那个是比较好的，完全是看你自己的网络环境而定喔！ OK！我们现在知道要连上以太网络组成的局域网络，就得要有网络卡、网络线、网络集中媒体(hub/switch)、 连上 Internet 的调制解调器等等，在这里鸟哥将防火墙、路由器等等设备归类为主机， 因为基本上，这些组件内部一定会含有一个网络卡，只是操作系统的精简程度与软件功能的不同就是了。 那么这些所需要的网络硬件又该如何挑选呢？

3.1.2 网络媒体选购建议

在开始底下的介绍之前，你必须要对于跳线、并行线、RJ-45 网络线、Hub/Switch 的优劣等等有一定程度的了解，请再前往第二章网络基础看一看。 此外，不在我们局域网络内的设备，例如调制解调器，那就得向你的 ISP 询问了！一般来说， 调制解调器是中华电信提供给用户的，然而由于『中华电信因为不同批次安装的调制解调器模块不同， 所以会有不一样的连接与线材处理方式！』(跳线与并行线的差异喔！) 所以请特别向你的 ISP 询问才行。底下主要针对局域网络内的网络媒体来进行介绍与说明。

- 主机硬件系统：考虑使用年限、省电、虚拟技术等

  过去我们都觉得旧的计算机拿来安装 Linux ，作为一个 Linux server 挺不错。后来鸟哥发现，很多旧计算机其实已经超过使用年限， 硬要使用有点问题，因为电子零件恐怕会撑不了太久的运作时间。而且，某些时刻生产的主机其实非常耗电！现在我们都强调要节能减碳嘛！ 所以，可以购买省电型的计算机主机，并且 CPU 含有虚拟化能力的更好。如此一来，不但比较省电，而且一部主机可以透过虚拟化的功能， 仿真出多部操作系统同时运作的环境，真正达到节能减碳的目的，那样也是很好的选择哟！

  不过，选购什么主机配备与该主机即将运作的服务其实是有关系的，例如防火墙系统与 DHCP 等服务并不需要很强的主机， 但是 Proxy 及 SQL 等服务器就得要强而有力的主机系统，甚至得要磁盘阵列的辅助会比较好！鸟哥在后续的章节所要介绍的服务， 大多仅是企业内部或者是外部很轻松的服务，并不需要什么强效的主机系统，因此目前的双核心入门级机种，已经非常棒了！ 所以啰，花太多时间在介绍主机硬件就变的没有什么意义！你只要记得，新购买主机时，最好选有伪指令集的 CPU 即可。

- Linux 操作系统：考虑稳定、可网络升级、能够快速取得协助支持

  你可以将目前的 distribution 分成两大类，一类是多功能新鲜货，例如 Fedora ， 一种是强调性能稳定但软件功能较旧的企业用途货，包括 RHEL, CentOS, SuSE 及 B2D 等！

一般来说，我们会建议你如果想要架设服务器时，尽量选择『稳定性较高的企业版』较佳， 因为功能新且强的版本例如 Fedora 由于太强调新鲜货，所以核心与软件的变动情况较为频繁， 那就很容易造成一些困扰，因为很多用户自行安装的软件可能无法在新的核心上面跑， 所以，只要核心一升级，哇！很多需要编译的软件就都需要再重新编译过！有点麻烦就是了。

由于鸟哥用惯了 RPM 以及 Red Hat 系统的关系，所以在这里推荐你使用 RHEL/CentOS/SuSE 这几个 Linux distributions，因为他够稳定且设定上面不难。不过，里面的软件版本可能就不会是最新的， 这点你可能就得要自行设法啰！比较特别的是 CentOS，他不但标榜完全相同于 RHEL ，并且可以直接透过 yum 这个软件进行完整版本的网络升级，既不会影响到原有的设定，升级时所花费的时间又短，所以， 目前鸟哥都是以这个版本来进行服务器的架设啊！

- 网络卡：考虑服务器用途、内建与否、驱动程序的取得等

一般来说，目前的新主机几乎都是内建 gigabit 的以太网络卡了，所以你不需要额外购买网络卡。 不过，使用内建的网络卡时，你得要注意到该网络卡是否为特殊的网络芯片，根据以往的经验，内建的网络卡通常是芯片较特殊的， 所以可能导致 Linux 预设的网卡驱动程序无法顺利的驱动该网络卡，那就比较累了～因为你必须要额外的安装网卡驱动程序之后， 才能够顺利的使用该款网络卡哩。

如果是想要作为 Linux 服务器的话，那么你的网络卡可能必须要购买好一点的。举例来说，某些主板内建便宜的 gigabit 网络接口，但越便宜的网络接口可能会造成损耗较多的 CPU 资源，如果能够购买类似 Intel/3Com 等知名品牌的 gigabit 适配卡，不但传输较为稳定，并且可以降低系统资源的耗费，是有一定程度的帮助的。另外，如果强调高速的话， 甚至可以选用 PCI-Express 的网络卡，而不使用传统的 PCI 接口。因为 PCI-Express 的传输带宽更高。

Tips: 你知道吗？鸟站 (http://linux.vbird.org) 使用的主机硬件是旧式的 AthlonXP 2000+， 内存也仅有 1.5G ，使用的网络卡则是早期的 3Com 3c905C 芯片，速度仅有 10/100 Mbps。但是，使用到目前流量传输是很顺畅的！ 不要说品牌迷思，有时产品的用料实在与否，很重要。




```
不过，如果是一般家用，或者是准备用来作为学习机之用的主机，那么万一网络卡芯片无法驱动时， 请先买个螃蟹卡 (
**Tips:**      如果要玩 Linux 又想比较顺畅的玩弄 Linux 时，请不要坚持使用 Linux 捉不到的网络卡！否则
```



- Switch/Hub：考虑主机数量、传输带宽、网管功能与否等

  就如同第二章网络基础里面曾经谈到的， Hub 是共享媒体而 Switch 是具有独立带宽的非共享媒体。因此以效能以及带宽来看，当然是 switch 比较好用啊！ 不过，如果你是一般家庭用户，只是要作简单的上网等工作，是没有必要购买太好的 switch 的， 建议使用一般大卖场可以买到的 5 port 的 hub 即可 (差不多 500 块台币的就不错了)。

  不过如果你常常在区网内传送大量的数据，例如一次传输就得要传送 GBytes 的数据时，那么网络的整体速度需要很详细的考虑喔！包括网络卡最好使用 gigabit ， 当然中间的联机设备最好买支持到 gigabit 速度的 switch 啦！因为 10/100/1000Mbps 的 switch 要比 10/100Mbps 的设备快上十倍，速度可是差很多的啊！如果你的设备还需要更快时， 例如鸟哥之前服务的实验室内部的 cluster (丛集式计算机群) ，则购买的 switch 甚至需要支持 Jumbo frame 这种支持大讯框的硬件架构才行，否则速度上不来啊！

- 网络线：考虑与速度相配的等级、线材形状、施工配线等

  在所有串连网络的设备当中，网络线是最重要，但是却也最容易被忽略～除了网络线的等级会影响到连接速度外，网络线所在处是否容易被压折？是否容易有讯号衰减？自己压制的 RJ-45 接头是否通过测试？网络线是否缠绕情况严重？都会影响到网络的传输优劣！所以，虽然我们常常讲要确认主机与 Switch 是否有连接成功可以看 switch 上的灯号，但是很多时候虽然灯号是亮的，不过由于网络线折损严重的问题， 也会导致联机质量不良喔！

  一般来说，『个体户』与小型企业通常网络线是直接放在外部的，这种情况你发现网络怪怪的时，可以直接更换线路。 不过，如果是如同中大型企业将网络线直接埋在墙内或者是在管线当中，发现问题时，真的很麻烦～ 所以才需要专业人才的辅助啊！

  Tips: 一般来说，越高等级的网络线，最好不要自行制作，因为一个小小的 RJ-45 接头的压制， 由于蕊线裸露程度的不同，就会影响到电子屏蔽效应的优劣了。Cat 5 等级的线材还可以自行压制， 比他还高等级的，最好还是买现成的吧！ ^_^



- 无线网络相关设备：考虑速度、标准、安全性等

现在的网络环境除了传统的有线网络之外，其实还有一个也是很常见的喔，那就是无线网络啦！ 无线网络会流行主要的原因除了笔记本电脑能力越来越强，使得很多朋友直接以笔记本电脑取代桌面计算机之外， 无线网络的速度目前已经可以达到 54 ~ 300 Mbps 那么快了 (802.11n 的标准而言)，对于一般只是上网看新闻与聊天的上班族来说， 这样的速度实在是非常快了 (一般的 ADSL 仅是 2M/256K bps 而已)， 所以要买无线网络设备 (含基地台与在 client 端的无线网卡) 来做成局域网络， 其实也是可以啦！而且还可以省去网络线的施工呢！

不过，无线网络最大的问题常常在于『无线的安全性』方面，因为是无线的设备， 所以『基地台如果没有做好防备措施的话，常常会导致 LAN 内的主机数据被窃取』， 这可是非常大的问题喔！可千万不要小看这个问题，吃上官司常常是由于忘记网络安全啊！ 记得购买无线网络基地台时，注意他可否『限制 MAC 』，如此一来，至少可以锁网卡，只让指定的网卡可以使用你的无线基地台，比较安全啦！

- 关于其他配件：

  事实上，整个网络环境可不止上头提到的这些咚咚而已，还包括硬件防火墙、路由器、网桥等等的， 当然，这些设备贵的话也有上百万的，但你的环境是否需要用到这么好的设备，那就见仁见智啊～此外， 为了环境的美观与生活的便利，你总不希望走在路上被网络线所绊倒，也不希望因为网络线绊倒你导致网络媒体掉落， 结果........损失了一堆 $$ 吧～所以啰，在网络线的转角处必须特别注意线材的保护， 在平面地上则需要特别使用压条给予固定，在牵线施工的时候尽量让线材沿着墙角或者是墙面上的既有物品， 如此则除了保持工作场所的美观之外，还能够增加工作场所的安全性啊！ ^_^

  此外，『计算机上网的速度并非完全取决于网络带宽』举例来说， 玩在线游戏时，大家都以为网络带宽需要很高规格，其实....根本不需要！因为 3D 联机游戏最主要的速度瓶颈应该是在于『3D 显示』而不是网络。这是因为网络仅传送一些数据给你的主机， 而你的主机再在自己的硬盘里面将图形取出，并且使用 3D 绘图卡将画面绘制到你的屏幕上。所以，显示速度或者是 CPU 不够力时，才会发生联机游戏的顿点。否则就是联机游戏服务器本身的负载 (loading) 太大，导致主机响应有较多延迟，就产生 lag (顿点) 的问题啦！

  另外，包括你主机使用的数据是否具有快速的传输接口也有关。举例来说，如果你的主机使用 USB 1.1 (最大传输 12Mbps)，但网络速度可达 10/100/1000Mbps ，那当你要在远程使用这部计算机的 USB 装置内的数据时，最大速度会是『 12Mbps 』，也就是最慢的那一个组件。所以啊， 网络速度慢的时候，不要以为只要增加网络带宽就好了，要确切的找出问题啊！

事实上，选购网络媒体所需要考虑的参数实在太多了，并且没有一定的依据，完全与使用者的使用环境与未来功能性有关。 不过，如果着眼在单纯的硬件速度上面的话，那么选购时考虑『我的网络速度可接受的最低速度为何？』去考虑吧！ 如果行有余力的话，再来考虑『我的环境需要多稳定的设备来达成？』其他的，那就得要靠你自己摸索啰！ ^_^

3.2 本书使用的内部联机网络参数与通讯协议

除非你已经具有相当熟练的 Linux 系统与服务器架设维护经验，否则不建议你使用上面图 3.1-4 所介绍的联机模式，对于初接触 Linux 服务器架设与维护的朋友来说，将你的联机模式设定成图 3.1-3 应该是个不错的选择，除了可以让你简单的就将服务器架设成功之外，也可以让你以 Linux 做为内部 LAN 的防火墙管理中心， 对于未来的学习成长方面较有帮助啊！ ^_^

3.2.1 联机参数与通讯协议

为使你的服务器学习之旅较有连贯性，因此鸟哥将后续章节会使用到的区网环境以图 3.1-3为范本， 设计一个区网，包括相关的网络参数如下所示：

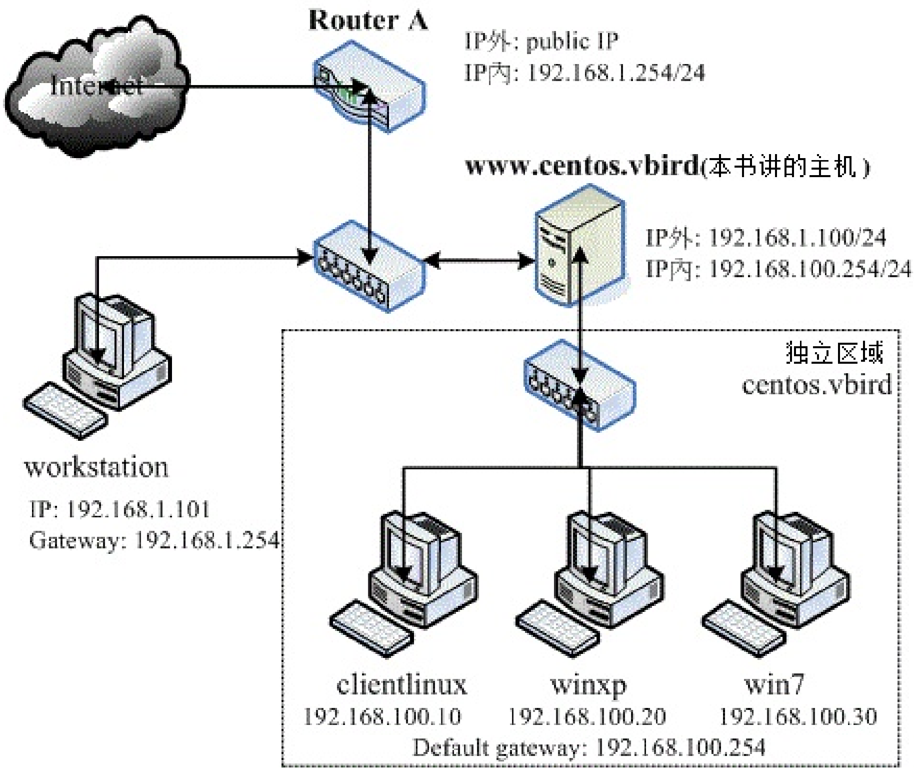


图 3.2-1、本书所使用的区网环境与参数设定

在上图的环境下，我们主要介绍的是 www.centos.vbird 那部 Linux 主机，该主机必须要具有 router 的能力，所以当然必须就要有两个接口，一个接口与 Internet 沟通，另一个接口则与内部的 LAN 沟通。那么为什么鸟哥说的是『两个网络接口』而不是『两张网络卡』呢？原因很简单， 因为一张网络卡可以设定多个 IP 啊！因此，在 Linux 当中一张网络卡可以具有一个以上的 IP 呢！由于一个 IP 即为一个网络接口，因此只要两个网络接口 (不论有几张网络卡) 即可进行 NAT (类似 IP 分享器功能) 的设定啦！所以自然一个网络卡即可啰！不过，鸟哥个人还是比较喜欢并且建议两张网络卡的啦， 将内外网络环境完整的分开，让你的内部网络效能较佳一点！

关于与 Internet 的联机方面，就如第二章谈到的，目前在台湾最常见的有 ADSL, Cable Modem, 学术网络的固定 IP 等， 这些联机的方式我们将在后续章节继续介绍的。至于内部的 LAN 我们则建议使用 Private IP 来设定喔！ 鸟哥通常喜欢使用 192.168.1.0/24 及 192.168.100.0/24 这几个 Class C 的网域，没什么特殊原因，只是因为....我喜欢！ ^_^ 在选定了 Private IP 的网段后，你必须要有『IP, Network, Netmask, Broadcast, Default gateway 以及 DNS 服务器的 IP』等等的设定值。 假设我 Linux 主机的对内 IP 为 192.168.100.254 ，则在图 3.2-1 内的 LAN 内的 PC 之网络相关设定参数则为：

- IP:设定为 192.168.100.1~192.168.100.253 ，但 IP 不可重复；
- Netmask：255.255.255.0
- Network：192.168.100.0、Broadcast：192.168.100.255
- Default Gateway：192.168.100.254 (路由器的 IP)
- DNS：暂时使用 168.95.1.1

Tips: 你能有如上图 3.2-1 这么多的计算机来测试你的服务器环境吗？当然不可能！那如何达成上述的功能呢？透过虚拟化技术啊！ 鸟哥是以 virtualbox 这套软件来处理整个局域网络所有主机的安装与测试，主要虚拟出来的机器有 www.centos.vbird, clientlinux, winxp, win7 这四部，要注意的是，www.centos.vbird 有两张网卡，一张为对外联机的使用 bridge 模式， 一张与其他三部主机联机的请选择『intnet』，这样就能搞定你的区网环境与实验喔！



- 安装什么通讯协议

目前网络社会最通用的通讯协议就是 TCP/IP 了！因此你如果要连上 Internet ，你的系统就得要支持 TCP/IP 才行。 但在局域网络内部时，事实上还可以透过简单的通讯协议来达到数据传输的目的，例如 NetBEUI 就是一个常见的简易通讯协议。

在 Linux 系统当中，只要将网络参数设定妥当，那么 TCP/IP 就已经被启用了，所以你不需要额外的再安装其他的通讯协议。不过，如果你需要将你的 Linux 系统中的硬盘空间分享给同网域的 Windows PC 时，那么就需要额外的加装 SAMBA 这个服务器软件才行。相关的 SAMBA 数据我们会在后面的章节提及。反正不管怎么说，目前 Internet 就是经由 TCP/IP 来进行连接的，而 Linux 本身就支持了 TCP/IP ，所以不需要额外的安装有的没的说！

至于在 Windows 部分就比较麻烦一点，因为在较大型的企业当中，还需要额外的考虑到 Windows Server 所提供的服务，那么在 Windows Clients 端就得要相应的启动某些通讯协议才行。一般来说，在 Windows Client 系统里面，最常见的两个通讯协议就是 TCP/IP 以及 NetBEUI 这两个通讯协议了。如果你只想让 Windows 与 Linux 能够藉由网络上的芳邻互通有无，那么启动 TCP/IP 也就够了 (因为 SAMBA 是藉由 NetBIOS over TCP/IP 来达成数据传输的) ，不过，也可以同时启动 NetBEUI 这个通讯协议就是了。

3.2.2 Windows 个人计算机网络设定范例

我们这本书谈论的是以 Linux 主机提供的服务器为主，所以关于 LAN 里面的 Windows 我都将他假设为 Client，并且不提供网络服务，所以都先以固定的 Private IP 来设定 Windows 操作系统，如果你的 LAN 有其他的考虑，那么底下的设定就看看就好。

我们在 Windows 系统上所需要的网络参数除了 IP, netmask, DNS 之外，还需要『工作组, workgroup』与 『计算机名称, Netbios name』等等的设定，此外，我们也可以加上 LAN 里面很常见的 NetBIOS (NetBEUI) 这个通讯协议呐。因此，除非你确定你的网域内还有其他的工作站， 否则『请只要安装 TCP/IP 以及 NetBEUI 这两个协议就好了！』 安装太多反而会有问题呢！底下我们假设你的网络卡都安装好了，并以图 3.2-1 里面那部内部区网的 winxp 主机为例来介绍：

- 与网络有关的设定参数：
    - IP: 192.168.100.20
    - Netmask: 255.255.255.0
    - DNS: 168.95.1.1
    - Gateway: 192.168.100.254
    - 工作组： vbirdhouse
    - 计算机名称： winxp
- 详细的设定流程：
    1. 先到『开始』==> 『设定』==> 『控制台』==> 『网络联机』==> 选择『区域联机』该项后，会出现如下图示：



图 3.2-2、区域联机状态

```
2.  上面画面当中选择『内容』进入如下的设定画面中：

    ![](img/winxp_net_02.gif)
    图 3.2-3、区域联机内容


3.  接下来，在上图中选择『联机后，将图标显示在通知区域内』， 并且双击『Internet Protocol (TCP/IP)』项

    ![](img/winxp_net_03.gif)
    图 3.2-4、区域联机 TCP/IP 设定内容


4.  再来需要编辑你的网络识别喔！选择『开始』==&gt; 『控制台』==&gt; 双击『系统』之后出现的图标，再按下

    ![](img/winxp_net_04.gif)
    图 3.2-5、区域联机计算机名称与工作组设定内容
```

基本上， Windows 的网络参数设定是相当的简单的！鸟哥这里仅介绍修改 IP 与相关网络参数的方式而已。 未来如果还需要搭配 DHCP 主机、NAT 主机等服务器的设定时，会再次的提醒用户 Windows 的设定信息喔！尤其是 SAMBA 主机的设定中， Windows 的网络识别就显的相当的重要呢！

2002/07/22：初次完成了局域网络架设的基本架构 2003/08/17：将整个文章重新修订，移除一些已经在 网络基础 里面谈过的内容，并且新增了表头的说明。 2003/08/20：加入课后练习了。 2003/09/19：加入参考用解答咯！ 2005/05/05：将原本介绍并行线与跳线的 N-Way 错误，订正为 auto MDI/MDI-x 2006/07/13：将旧的文章移动到 此处 。 2006/07/14：加入 Linux distribution 的说明，并且重新校稿更新内容，如布线部分。 2006/07/15：取消 Windows

2000, 98。 2010/08/16：将原本旧的基于 CentOS 4.x 的版本移动于此处 2010/08/20：由于很多数据与前一章有重复，有些数据太过老旧，所以重新编辑过此文章。 2011/07/15：将原本的基于 CentOS 5.x 的文章移动到此处 2011/07/15：最重要的是加入图 3.2-1 了！那与服务器篇后续所有的章节都有相关性喔！

第四章、连上 Internet

最近更新日期：2011/07/20

终于要来到修改 Linux 网络参数的章节了！在第二章的网络基础中， 我们知道主机要连上 Internet 需要一些正确的网络参数设定，这些设定在 Windows 系统上面的修改则在第三章的局域网络架构中说明了。在这一章当中，我们则主要以固定 IP 的设定方式来修改 Linux 的网络参数，同时，也会介绍如何使用 ADSL 的拨接方式来上网，此外，因为 Cable modem 使用者也不在少数，所以我们也说明一下 Cable modem 在 Linux 下的设定方式喔！ 最后，由于笔记本电脑使用者大增，且因为笔记本电脑常使用无线网络，因此本文也加入了无线网络的联机介绍啊！

4.1 Linux 连上 Internet 前的注意事项

由前面几章的数据我们知道，想要连上 Internet 你得要设定一组合法的 IP 参数才可以，主要是 IP, Netmask, Gateway, DNS IP 以及主机名等。那我们也知道，其实整个主机最重要的设定，就是『先要驱动网络卡』，否则主机连网络卡都捉不到时， 怎么设定 IP 参数都是没有用的，你说是吧！所以底下我们就来谈一谈， 你要如何确定网络卡已经被捉到，而 Linux 主机的网络参数又该如何设定？

4.1.1 Linux 的网络卡

你怎么确认 Linux 有捉到网络卡？Linux 底下的网络卡的名称是啥？让我们来了解一下吧！

- 认识网络卡的装置代号

  在 Linux 里面的各项装置几乎都是以文件名来取代的，例如 /dev/hda 代表 IDE1 接口的第一个 master 硬盘等等。 不过，网络卡的代号 (Network Interface Card, NIC) 却是以模块对应装置名称来代替的， 而默认的网络卡代号为 eth0 ，第二张网络卡则为 eth1 ，以此类推。

- 关于网络卡的模块 (驱动程序)

  我们知道网络卡其实是硬件，所以当然需要核心支持才能驱动他。一般来说，目前新版的 Linux distributions 默认可以支持的网络卡芯片组数量已经很完备了，包括大厂的 3COM, Intel 以及初阶的 RealTek, D-Link 等网络卡芯片都已经被支持， 所以使用者可以很轻易的设定好他们的网络卡。不过，万一你的网络卡芯片组开发商不愿意释出开放源 (Open Source) 的硬件驱动程序，或者是该网络卡太新了，使得 Linux 核心来不及支持时，那么你就得要透过：

  1. 重新编译较新的核心，或者是
  2. 编译网络卡的核心模块 好让核心可以支持网络卡这块硬件啦！但是，重编核心或编译网络卡核心模块都不是简单的工作， 而且有时原始码又可能无法在每部主机上面编译成功，所以万一你的网络卡真的不被默认的 Linux 网络芯片所支持，那么鸟哥真的建议直接换一块被 Linux 支持的网络卡吧，例如很便宜的螃蟹卡！ 免得花了太多时间在硬件确认上面，划不来的！ ^_^

  另外，其实有的时候 Linux 的默认网络卡模块可能无法完全 100% 的发挥网络卡的功能的， 所以，有的时候你还是得必须要自行编译网络卡的模块才行喔！当然， 那个网络卡的模块就得要自行由网络卡开发商的官方网站下载了！不过，如果你的网络卡是自行编译安装的， 那么每次重新安装其他版本的核心时，你都必须要自行重新手动编译过该模块。 因为模块与核心是有相关性的啊！

- 观察核心所捉到的网卡信息

假设你的网络卡已经在主机上面，不论是内建的还是自行安插到 PCI 或 PCI-x 或 PCI-E 的接口上， 那么如何确认该网络卡有被核心捉到呢？很简单啊！就利用 dmesg 来查阅即可：

```
[root@www ~]# dmesg &#124; grep -in eth
377:e1000: eth0: e1000_probe: Intel(R) PRO/1000 Network Connection
383:e1000: eth1: e1000_probe: Intel(R) PRO/1000 Network Connection
418:e1000: eth0 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX
419:eth0: no IPv6 routers present
```

从上面的第 377 及 383 这两行，我们可以查到这部主机的两张网络卡都使用模块为 e1000，而使用的芯片应该就是 Intel 的网络卡了。此外，这个网卡的速度可达到 1000Mbps 的全双工模式哩 (418行) ！除了使用 dmesg 来查询核心侦测硬件产生的信息外，我们也可以透过 lspci 来查询相关的设备芯片数据喔！如下所示：

```
[root@www ~]# lspci &#124; grep -i ethernet
00:03.0 Ethernet controller: Intel Corporation 82540EM Gigabit Ethernet
Controller (rev 02)
```

请注意，鸟哥这里使用的是 Virtualbox 仿眞的那部主机的环境 (请参考第一章 1.2.2-2)，因此使用的是模拟出来的 Intel 网卡。 如果你是使用自己的实际硬件配备安装的主机，那么应该会看到不同的芯片啦！那是正常的！

- 观察网络卡的模块

从刚刚的 dmesg 的输出讯息中，我们知道鸟哥这部主机所使用的模块是 e1000 ，那核心有顺利的载入了吗？可以利用 lsmod 去查查看。此外，这个模块的相关信息又是如何呢？使用 modinfo 来查查看吧！

```
[root@www ~]# lsmod &#124; grep 1000
e1000                 119381  0  &lt;==确实有载入到核心中！

[root@www ~]# modinfo e1000
filename:    &lt;u&gt;/lib/modules/2.6.32-71.29.1.el6.x86_64/kernel/drivers/net/e1000
version:     7.3.21-k6-NAPI
license:     GPL
description: Intel(R) PRO/1000 Network Driver
.....(以下省略).....
```

上面输出信息的重点在于那个档名 (filename) 的部分！那一场串的文件名目录，就是我们驱动程序放置的主要目录所在。 得要注意的是，那个 2.6.32-71.29.1.el6.x86_64 是核心版本，因此，不同的核心版本使用的驱动程序其实不一样喔！我们才会一直强调，更改核心后， 你自己编译的硬件驱动程序就需要重新编译啦！

那你如何知道你的网络卡卡号呢？很简单啊！不管有没有启动你的网络卡，都可以使用：『 ifconfig eth0 』来查询你的网卡卡号。如果你照着上面的信息来作， 结果发现网卡已经驱动了，恭喜你，准备到下一节去设定网络吧！如果没有捉到网卡呢？那就准备

自己编译网卡驱动程序吧！

4.1.2 编译网卡驱动程序(Option)

一般来说，如果没有特殊需求，鸟哥不是很建议你自己编译网络卡的驱动程序！为啥？ 因为想到每次更新核心都得要重新编译一次网卡驱动程序，光是想想都觉得烦～所以，没有被 Linux 预设核心支持的网卡， 就先丢着吧！

Tips: 鸟哥之前买了一张内建网卡的主板，该网卡并没有被当时的 Linux 预设核心所支持，所以就得要自己编译核心啦。因为 CentOS 很少更新核心，所以第一次编译完毕之后就忘记有这回事了。等到过了数周有新的核心出现后，鸟哥很开心的自动升级核心， 然后远程进行 reboot ，结果呢？没有网卡驱动程序了啦！我的主机无法联网，得要到主机前用 tty 登入后才能进行编译～唉～



如果你真的很有求知欲，而且该网卡的官网有提供给 Linux 的驱动程序原始码； 或者是你很想要某些官网提供的驱动程序才有的特殊功能；又或者是你真的很不想要再买一张额外的网卡。 此时，就得要重新编译网络卡的驱动程序啰。

Tips: 事实上，如果你要新添购硬件时，请先查阅一下硬件包装上面是否提及支持 Linux 的字样， 因为有些硬件厂商在推出新硬件时，常常会漏掉 Linux 驱动程序的撰写。 如果包装上面有提到支持的话，那么至少你会获得官方网站所提供的驱动程序原始码啊！ ^_^



因为我们这里使用的网络卡是 Intel 的 82540EM Gigabit Ethernet 控制芯片，假设你需要的驱动程序得要由 Intel 官网取得最新的版本，而不要使用预设的核心所提供的版本时，那你该如何处理呢？请注意，鸟哥这个小节只是一个范例简介， 不同的厂商推出的驱动程序安装方式都有点不太一样，你得要参考驱动程序的读我档 (READ ME) 或相关档案来安装才行。 此外，如果默认驱动程序已经捉到了网络卡，鸟哥是建议使用预设的驱动程序就好了喔！

另外，由于编译程序需要编译程序以及核心相关信息，因此得要预安装 gcc, make, kernel-header 等软件才行。 但是我们选择的安装模式为『basic server』，这些软件默认都没有安装的，所以你得要先安装这些软件才行。 这些软件可以简单的透过 yum 使用『yum groupinstall 'Development Tools'』来安装，只可惜你并没有网络啊！ 所以就得要透过原本光

盘一个一个去处理 RPM 属性相依的问题来解决了～很麻烦的～不然的话，就得要透过更改 yum 配置文件，使用本机档案的类型来取得原版光盘的 yum 软件列表啰！鸟哥这里假设你已经安装了所需要的编译程序了， 接下来的动作是：

1. 取得官方网站的驱动程序：

   再次说明，你可以复制鸟哥的环境，透过 Virtualbox 的模拟而来。我们这里使用的是 Intel 的网卡，你可以到如下的网站去下载：

   - http://downloadcenter.intel.com/SearchResult.aspx?lang=eng&keyword='e1000-' 最后 (2011/07) 下载的版本为 8.0.30，确实比上个小节提到的版本还要新！下载的文件名为 e1000-8.0.30.tar.gz， 鸟哥将它放置于 /root 底下，然后准备来处理编译过程吧！

2. 解压缩与编译：

   使用 root 的身份进行如下工作吧：

```
[root@www ~]# tar -zxvf e1000-8.0.30.tar.gz -C /usr/local/src
[root@www ~]# cd /usr/local/src/e1000-8.0.30/
# 此时在该目录下有个 README 的档案，记得看一看，这个档案内会说明很多信息，
# 包括如何编译，以及这个模块所支持的芯片组哩！
[root@www e1000-8.0.30]# cd src
[root@www src]# make install
```

   最后这个模块会被编译完成且安装放置于如下的档名： /lib/modules/$(uname -r)/kernel/drivers/net/e1000/e1000.ko。 接下来我们得要重载这个新的模块才行哟！

3. 模块之测试与处理

   由于这个模块已经被加载啦，所以我们得要先移除旧的模块后，才能够重载这个模块。使用的方法有点像这样：

```
# 1\. 先移除已经加载在内存中的旧模块
[root@www ~]# rmmod e1000
# 此时已经捉到的网卡会整个消失不见！因为驱动程序被卸除了嘛！

# 2\. 加载新模块，并且查阅一下有没有捉到正确的版本！
[root@www ~]# modprobe e1000
[root@www ~]# modinfo e1000
filename:       /lib/modules/2.6.32-71.29.1.el6.x86_64/kernel/drivers/net/e1000/e1000
version:        &lt;u&gt;8.0.30-NAPI&lt;/u&gt;  &lt;==就是这里！
license:        GPL
description:    Intel(R) PRO/1000 Network Driver
```

   请自行与前一小节比对一下，就会发现真的捉到正确的版本啰！不过，这个模块在下次新的核心推出后就会失效！ 为什么呢？因为新核心会给一个新的驱动程序嘛！就不是你现在这个 8.0.30 的版本啰。这点还是要再次说明的。

4. 设定开机自动启动网络卡模块 (Option)

如果你在开机就能够正确的取得这个模块的话，那么你的网卡就没有问题啦！这个步骤是可以略过的。如果你的核心还是捉不到网卡， 那你可能得要自己处理一下模块的对应才行。怎么处理呢？很简单，在 /etc/modprobe.d/ 目录下建立一个名为 ether.conf 的档案， 内容将模块与网卡代号链接在一块即可！这样处理：

```
[root@www ~]# vim /etc/modprobe.d/ether.conf
alias eth0 e1000
alias eth1 e1000  &lt;==因为鸟哥有两张网卡嘛！

[root@www ~]# sync; reboot
```

为了测试一下刚刚的设定是否会生效，通常鸟哥都会尝试一次重新启动，然后开机完成之后观察一下是否有正确的启动网络卡， 并观察一下模块加载的情况，如果一切都顺利，那就太完美了！

5. 尝试设定 IP

等到一切就绪之后，总得试看看这样的网络卡模块是否可以顺利的设定好 IP 吧？ 所以我们先手动给他一个私有 IP 看看先：

```
[root@www ~]# ifconfig eth0 192.168.1.100
[root@www ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:71:85:BD
          inet addr:192.168.1.100  Bcast:192.168.1.255  Mask:255.255.255.0
....(以下省略)....
```

嘿嘿！真的设定妥当哩！然后利用 ping 这个指令去 ping 一下网域内的其他计算机， 看看能不能有响应，就知道你的网络卡是否 OK 的啦！通常是没有问题的啦！

4.1.3 Linux 网络相关配置文件案

我们知道 TCP/IP 的重要参数主要是： IP, Netmask, Gateway, DNS IP ，而且千万不要忘记你这部主机也应该要有主机名 (hostname)！此外，我们也知道 IP 的取得有手动设定、DHCP 处理等。那么这些参数主要是写在哪些配置文件？如何对应呢？ 底下就让我们来处理一番！

所需网络参数	主要配置文件档名	重要参数
IP Netmask DHCP 与否 Gateway 等	/etc/sysconfig/network-scripts/ifcfg-eth0	DEVICE=网卡的代号 BOOTPROTO=是否使用dhcp HWADDR=是否加入网卡卡号(MAC) IPADDR=就是IP地址 NETMASK=只网络屏蔽啦 ONBOOT=要不要默认启动此接口 GATEWAY=就是通讯闸啦 NM_CONTROLLED=额外的网管软件 鸟哥建议取消这个项目！
主机名	/etc/sysconfig/network	NETWORKING=要不要有网络 NETWORKING_IPV6=支持IPv6否？ HOSTNAME=你的主机名
DNS IP	/etc/resolv.conf	nameserver DNS的IP
私有 IP 对应的主机名	/etc/hosts	私有IP 主机名 别名

你没有看错，主要需要修改的就是这四个档案而已！因此没有很困难！大家都想太多了！详细的设定后续小节再来讲， 这里先有概念即可。除此之外，还有些档案或许你也应该要知道一下比较好哟！

- /etc/services 这个档案则是记录架构在 TCP/IP 上面的总总协议，包括 http, ftp, ssh, telnet 等等服务所定义的 port number ，都是这个档案所规划出来的。如果你想要自定义一个新的协议与 port 的对应，就得要改这个档案了；
- /etc/protocols 这个档案则是在定义出 IP 封包协议的相关数据，包括 ICMP/TCP/UDP 这方面的封包协议的定义等。

大概知道上面这几个档案后，未来要修改网络参数时，那就太简单了！至于网络方面的启动指令的话，可以记得几个简单的指令即可喔！

- /etc/init.d/network restart 这个 script 最重要！因为可以一口气重新启动整个网络的参数！ 他会主动的去读取所有的网络配置文件，所以可以很快的恢复系统默认的参数值。
- ifup eth0 (ifdown eth0) 启动或者是关闭某张网络接口。可以透过这个简单的 script 来处理喔！ 这两个 script 会主动到 /etc/sysconfig/network-scripts/ 目录下， 读取适当的配置文件来处理啊！ (例如 ifcfg-eth0)。

大概你只要只到这些基本的指令与档案，哈哈！网络参数的设定就太简单啦！ 不过，最好你还是要了解 shell script ，比较好！因为可以追踪整个网络的设定条件。 why ？这是因为每个 distributions 的设定数据可能都不太相同，不过却都以 /etc/init.d/network 作为启动的 script ，因此，你只要了解到该档案的内容，很容易就追踪得出来你的配置文件所需要的内容呢！对吧！

另外，新版的 CentOS 6.x 还有额外推出一个名称为 NetworkManager 的软件机制来管理网络， 不过，鸟哥还是比较喜欢手工打造自己的网络环境，所以很建议将该软件关闭呢！还好，我们安装的『basic server (第一章的 1.2.2-2)』 就这么巧的没有安装该软件！好佳在～ ^_^

4.2 连上 Internet 的设定方法

在前几章我们就谈过，台湾地区主要连上因特网的方法有(1)学术网络、(2)ADSL 固接与拨接、(3)Cable modem 等方式，同时，手动设定 IP 参数是很重要的学习，因此，底下的各节中，第一节的手动设定固定 IP 一定要做过一次！ 其他的才依照您的环境去设定去学习！

此外，由于目前使用 Linux notebook 的使用者大增，而 Notebook 通常是以无线网络来联机的， 所以鸟哥在这里也尝试使用一款无线网络来进行联机设定。至于传统的 56 Kbps 拨接则因为速度较慢且使用度越来越低， 所以在这里就不多做介绍了。

4.2.1 手动设定固定 IP 参数 (适用学术网络、ADSL 固定制) + 五大检查步骤

所谓的固定 IP 就是指在你的网络参数当中，你只要输入既定的 IP 参数即可。那么这个既定的 IP 来自哪里呢？ 一般来说，他可能来自于：

- 学术网络：由学校单位直接给予的一组 IP 网络参数；
- 固定制 ADSL：向 ISP 申请的一组固定 IP 的网络参数；
- 企业内部或 IP 分享器内部的局域网络：例如企业内使用私有 IP 作为局域网络的联机之用时，那么我们的 Linux 当然也就需要向企业的网管人员申请一组固定的 IP 网络参数啰！

这样清楚吗？也就是说，我们取得的固定 IP 参数并非一定是 public IP 喔！反正就是一组可接受的固定 IP 就是了！ 所以在架设你的环境之前，请先注意所有网络参数的来源正确性啊！好了，那么你的 IP 要如何设定呢？先回去翻翻第三章 3.2.1 里面的图 3.2-1，我们对外网卡 (eth0) 的信息为：

```
IP:       192.168.1.100
Netmask:  255.255.255.0
Gateway:  192.168.1.254
DNS IP:   168.95.1.1
Hostname: www.centos.vbird
```

那么要修改的四个档案与相关的启动脚本，以及重新启动后需要用啥指令观察的重点，鸟哥再次的使用一个简单的表格来说明， 你只要记得这几个表格内的重要档案与指令，以后在修改网络参数时，就不会出现错误了！看看吧！

修改的参数	配置文件与重要启动脚本	观察结果的指令
IP相关参数	/etc/sysconfig/network-scripts/ifcfg-eth0 /etc/init.d/network restart	ifconfig (IP/Netmask) route -n (gateway)
DNS	/etc/resolv.conf	dig www.google.com
主机名	/etc/sysconfig/network /etc/hosts	hostname (主机名) ping $(hostname) reboot

底下我们就分别针对上面的各项设定来进行档案的重新修改啰！

1. IP/Netmask/Gateway 的设定、启动与观察

   设定网络参数得要修改 /etc/sysconfig/network-scripts/ifcfg-eth0，请记得，这个 ifcfg-eth0 与档案内的 DEVICE 名称设定需相同，并且，在这个档案内的所有设定，基本上就是 bash 的变量设定规则啦 (注意大小写)！

```
[root@www ~]# vim /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE="eth0"                &lt;==网络卡代号，必须要 ifcfg-&lt;u&gt;eth0&lt;/u&gt; 相对应
HWADDR="08:00:27:71:85:BD"   &lt;==就是网络卡地址，若只有一张网卡，可省略此项目
NM_CONTROLLED="no"           &lt;==不要受到其他软件的网络管理！
ONBOOT="yes"                 &lt;==是否默认启动此接口的意思
BOOTPROTO=none               &lt;==取得IP的方式，其实关键词只有dhcp，手动可输入none
IPADDR=192.168.1.100         &lt;==就是 IP 啊
NETMASK=255.255.255.0        &lt;==就是子网掩码
GATEWAY=192.168.1.254        &lt;==就是预设路由
# 重点是上面这几个设定项目，底下的则可以省略的啰！
NETWORK=192.168.1.0          &lt;==就是该网段的第一个 IP，可省略
BROADCAST=192.168.1.255      &lt;==就是广播地址啰，可省略
MTU=1500                     &lt;==就是最大传输单元的设定值，若不更改则可省略
```

   上面的资料很好理解吧！请注意每个变量(左边的英文)都应该要大写！ 否则我们的 script 会误判！事实上鸟哥的设定值只有最上面的 8 个而已，其他的 NETWORK, BROADCAST, MTU 鸟哥都没有设定喔！ 至于参数的说明方面，IPADDR, NETMASK, NETWORK, BROADCAST 鸟哥在这里就不再多说，要谈的是几个重要的设定值：

   - DEVICE：这个设定值后面接的装置代号需要与文件名 (ifcfg-eth0) 那个装置代号相同才行！否则可能会造成一些装置名称找不到的困扰。
   - BOOTPROTO：启动该网络接口时，使用何种协议？ 如果是手动给予 IP 的环境，请输入 static 或 none ，如果是自动取得 IP 的时候， 请输入 dhcp (不要写错字，因为这是最重要的关键词！)
   - GATEWAY：代表的是『整个主机系统的 default gateway』， 所以，设定这个项目时，请特别留意！不要有重复设定的情况发生喔！也就是当你有 ifcfg-eth0, ifcfg-eth1.... 等多个档案，只要在其中一个档案设定 GATEWAY 即可

- GATEWAYDEV：如果你不是使用固定的 IP 作为 Gateway ， 而是使用网络装置作为 Gateway (通常 Router 最常有这样的设定)，那也可以使用 GATEWAYDEV 来设定通讯闸装置呢！不过这个设定项目很少使用就是了！

- HWADDR：这个东西就是网络卡的卡号了！在仅有一张网卡的情况下，这个设定值没有啥功能， 可以忽略他。但如果你的主机上面有两张一模一样的网卡，使用的模块是相同的。 此时，你的 Linux 很可能会将 eth0, eth1 搞混，而造成你网络设定的困扰。如何解决呢？ 由于 MAC 是直接写在网卡上的，因此指定 HWADDR 到这个配置文件中，就可以解决网卡对应代号的问题了！很方便吧！ 设定完毕之后，现在让我们来重新启动网络接口吧！这样才能更新整个网络参数嘛！

```
[root@www ~]# /etc/init.d/network restart
Shutting down interface eth0:          [ OK ]  &lt;== 先关闭界面
Shutting down loopback interface:      [ OK ]
Bringing up loopback interface:        [ OK ]  &lt;== 再开启界面
Bringing up interface eth0:            [ OK ]
# 针对这部主机的所有网络接口 (包含 lo) 与通讯闸进行重新启动，所以网络会停顿再开
```

这样就处理完毕啰，那接下来当然就是观察看看啰！

```
# 检查一：当然是要先察看 IP 参数对否，重点是 IP 与 Netmask 啦！
[root@www ~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 08:00:27:71:85:BD
          inet addr:192.168.1.100  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe71:85bd/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:655 errors:0 dropped:0 overruns:0 frame:0
          TX packets:468 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:61350 (59.9 KiB)  TX bytes:68722 (67.1 KiB)
# 有出现上头那个 IP 的数据才是正确的启动；特别注意 inet addr 与 Mask 项目
# 这里如果没有成功，得回去看看配置文件有没有错误，然后再重新 network restart ！

# 检查二：检查一下你的路由设定是否正确
[root@www ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
0.0.0.0         192.168.1.254   0.0.0.0         UG    0      0        0 eth0
# 重点就是上面的特殊字体！前面的 0.0.0.0 代表预设路由的设定值！

# 检查三：测试看看与路由器之间是否能够联机成功呢！
[root@www ~]# ping -c 3 192.168.1.254
PING 192.168.1.254 (192.168.1.254) 56(84) bytes of data.
64 bytes from 192.168.1.254: icmp_seq=1 ttl=64 time=2.08 ms
64 bytes from 192.168.1.254: icmp_seq=2 ttl=64 time=0.309 ms
64 bytes from 192.168.1.254: icmp_seq=3 ttl=64 time=0.216 ms

--- 192.168.1.254 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 0.216/0.871/2.088/0.861 ms
# 注意啊！有出现 ttl 才是正确的响应！如果出现『 Destination Host Unreachable 』
# 表示没有成功的联机到你的 GATEWAY 那表示出问题啦！赶紧检查有无设定错误。
```

要注意，第三个检查如果失败，可能要看你的路由器是否已经关闭？或者是你的 switch/hub 是否有问题，或者是你的网络线是否错误，还是说你的或路由器的防火墙设定错误了？要记得去解决喔！ 这三个检查做完而且都成功之后，那么你的 TCP/IP 参数设定已经完毕了！这表示你可以使用 IP 上网啦！ 只是还不能够使用主机名上网就是了。接下来就是要设定 DNS 啰！

2. DNS 服务器的 IP 设定与观察

这个 /etc/resolv.conf 很重要啦！他会影响到你是否可以查询到主机名与 IP 的对应喔！通常如下的设定就 OK 了！

```
[root@www ~]# vim /etc/resolv.conf
nameserver 168.95.1.1
nameserver 139.175.10.20
```

我们以中华电信与 SeedNet 在南部的 DNS 服务器之 IP 作为设定的方式！请注意一下，如果你不知道你的最接近的 DNS 服务器的 IP ，那么直接输入 nameserver 168.95.1.1 这个中华电信的 DNS 主机即可！不过如果你公司内部有设定防止 DNS 的要求封包的防火墙规则时，那么你就得要请教贵公司的网管单位告知你的 DNS IP 设定啦！然后赶紧测试看看：

```
# 检查四：看看 DNS 是否顺利运作了呢？很重要的测试喔！
[root@www ~]# dig www.google.com
....(前面省略)....
;; QUESTION SECTION:
;www.google.com.                        IN      A

;; ANSWER SECTION:
www.google.com.         428539  IN      CNAME   www.l.google.com.
www.l.google.com.       122     IN      A       74.125.71.106
....(中间省略)....

;; Query time: 30 msec
;; SERVER: 168.95.1.1#53(168.95.1.1)  &lt;==这里的项目也很重要！
;; WHEN: Mon Jul 18 01:26:50 2011
;; MSG SIZE  rcvd: 284
```

上面的输出有两个重点，一个是问题查询的是 www.google.com 的 A (Address) 参数，并且从回答 (Answer) 里面得到我们所需的 IP 参数。最后面一段的 Server 项目非常重要！你得要看是否与你的设定相同的那部 DNS 服务器 IP 才行！ 以上面输出为例，鸟哥使用中华电信的 DNS 服务器，所以就出现 168.95.1.1 的 IP 地址啰。

3. 主机名的修改、启动与观察

修改主机名就得要改 /etc/sysconfig/network 以及 /etc/hosts 这两个档案，这两个档案的内容又简单的要命喔！

```
[root@www ~]# vim /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=www.centos.vbird

[root@www ~]# vim /etc/hosts
192.168.1.100    www.centos.vbird
# 特别注意，这个档案的原本内容不要删除！只要新增额外的数据即可！
```

修改完毕之后要顺利启动的话，得要重新启动才可以。为什么需要重新启动呢？因为系统已经有非常多的服务启动了， 这些服务如果需要主机名，都是到这个档案去读取的。而我们知道配置文件更新过后，服务都得要重新启动才行。 因此，已经启动而且有读到这个档案的服务，就得要重新启动啊！真麻烦～因此，最简单的方法，就是重新启动。但重开机之前还需要进行一项工作，否则，你的系统开机会花掉很多时间喔！

```
[root@www ~]# hostname
localhost.localdomain
# 还是默认值，尚未更新成功！我们还得要进行底下的动作！

# 检查五：看看你的主机名有没有对应的 IP 呢？没有的话，开机流程会很慢！
[root@www ~]# ping -c 2 www.centos.vbird
PING www.centos.vbird (192.168.1.100) 56(84) bytes of data.
64 bytes from www.centos.vbird (192.168.1.100): icmp_seq=1 ttl=64 time=0.015 ms
64 bytes from www.centos.vbird (192.168.1.100): icmp_seq=2 ttl=64 time=0.028 ms

--- www.centos.vbird ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.015/0.021/0.028/0.008 ms
# 因为我们有设定 /etc/hosts 规定 www.centos.vbird 的 IP ，
# 所以才找的到主机主机名对应的正确 IP！这时才能够 reboot 喔！重要重要！
```

上面的信息中，检查的内容总共有五个步骤，这五个步骤每一步都要成功后才能够继续往下处理喔！ 至于最重要的一点，当你修改过 /etc/sysconfig/network 里面的 HOSTNAME 后，务必要重新启动 (reboot)。但是重新启动之前，请务必『 ping 主机名』且得到 time 的响应才行！

4.2.2 自动取得 IP 参数 (DHCP 方法，适用 Cable modem、IP 分享器的环境)

可自动取得 IP 的环境是怎么回事啊？不是很简单吗？当你在 IP 分享器后头的主机在设定时，不是都会选择『自动取得 IP 』吗？那就是可自动取得 IP 的环境啦！那么这个自动取得是怎么回事啊？也不难了解啦，其实就是『有一部主机提供 DHCP 服务给整个网域内的计算机』就是了！例如 IP 分享器就可能是一部 DHCP 主机。那么 DHCP 是啥？ 他是：Dynamic Host Configuration Protocol 的简写，顾名思义，他可以『动态的调整主机的网络参数』的意思。详细的 DHCP 功能我们会在第十二章说明的。好了，那么这个方法适合哪些联机的方式呢？大致有这些：

- Cable Modem：就是使用电视缆线进行网络回路联机的方式啊！
- ADSL 多 IP 的 DHCP 制：就鸟哥所知， SeedNet 有推出一种专案， 可以让 ADSL 用户

以 DHCP 的方式来自动取得 IP ，不需要拨接。那使用的也是这种方法！

- IP 分享器或 NAT 有架设 DHCP 服务时：当你的主机位于 IP 分享器的后端，或者是你的 LAN 当中有 NAT 主机且 NAT 主机有架设 DHCP 服务时， 那取得 IP 的方法也是这样喔！

你依旧需要前一小节手动设定 IP 的主机名设定 (第三步骤)，至于 IP 参数与 DNS 则不需要额外设定， 仅需要修改 ifcfg-eth0 即可喔！这样处理吧：

```
[root@www ~]# vim /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
HWADDR="08:00:27:71:85:BD"
NM_CONTROLLED="no"
ONBOOT=yes
BOOTPROTO=dhcp
```

没盖你喔！只要这几个项目即可，其他的都给他批注 (#) 掉！尤其是那个 GATEWAY 一定不能设定！ 避免互相干扰！然后给他重新启动网络：

```
[root@www ~]# /etc/init.d/network restart
Shutting down interface eth0:          [ OK ]  &lt;== 先关闭界面
Shutting down loopback interface:      [ OK ]
Bringing up loopback interface:        [ OK ]  &lt;== 再开启界面
Bringing up interface eth0:            [ OK ]
Determining IP information for eth0.. [ OK ]  &lt;== 重要！是 DHCP 的特点！
# 你可以透过最后一行去判断我们是否有透过 DHCP 协议取得 IP！
```

我们局域网络内的 IP 分享器或 DHCP 主机，就会立刻帮你的 Linux 主机做好网络参数的规划， 包括 IP 参数与 GATEWAY 等，就通通设定妥当啦！很方便也很简单吧！

Tips: 基本上，/etc/resolv.conf 预设会被 DHCP 所修改过，因此你不需要修改 /etc/resolv.conf。甚至连主机名都会被 DHCP 所修订。不过，如果你有特殊需求，那么 /etc/sysconfig/network 以及 /etc/hosts 请自行修改正确哟！



4.2.3 ADSL 拨接上网 (适用台湾 ADSL 拨接以及光纤到大楼)

终于来到台湾最热门的 ADSL 拨接上网的介绍啦！来谈一谈如何在 Linux 上拨接上网吧！ 要拨接上网时，可以使用 rp-pppoe 这套软件来帮忙(注1)，所以，你必须要确认你的 Linux distributions 上面已经存在这个玩意儿了！CentOS 本身就含有 rp-pppoe ，请使用原版光盘，或者是使用 yum 来进行安装吧！

```
[root@www ~]# mount /dev/cdrom /mnt
[root@www ~]# cd /mnt/Packages
[root@www ~]# rpm -ivh rp-pppoe* ppp*
[root@www ~]# rpm -q rp-pppoe
rp-pppoe-3.10-8.el6.x86_64    &lt;==你瞧瞧！确实已经安装喔！
```

当然，很多 distributions 都已经将拨接这个动作归类到图形接口里面去了，所以可能没有提供 rp-pppoe 这个咚咚，没关系，你可以到底下的网站去取得的：

- http://www.roaringpenguin.com/pppoe/
- http://freshmeat.net/projects/rp-pppoe/

然后再自行手动安装即可。如何安装的过程鸟哥在这里就不谈了，请自行前往基础篇的原始码与 Tarball 章节查阅相关资料吧。 另外请注意，虽然整个联机是由主机的以太网络卡连接到 ADSL 调制解调器上，然后再透过电话线路联机到 ISP 的机房去，最后在主机上以 rp-pppoe 拨接达成联机。但是 rp-pppoe 使用的是 Point to Point (ppp) over Ethernet 的点对点协议所产生的网络接口，因此当你顺利的拨接成功之后， 会多产生一个实体网络接口『 ppp0 』喔！

而由于 ppp0 是架构在以太网络卡上的，你必须要有以太网卡，同时，即使拨接成功后，你也不能将没有用到的 eth0 关闭喔！注意注意！因此，拨接成功后就会有：

- 内部循环测试用的 lo 接口；
- 网络卡 eth0 这个接口；
- 拨接之后产生的经由 ISP 对外连接的 ppp0 接口。

虽然 ppp0 是架构在以太网卡上面的，但上头这三个接口在使用上是完全独立的，互不相干，所以关于 eth0 的使用上，你就可以这样思考：

- 这张网络卡 (假设是 eth0) 有接内部网络(LAN)：

  举例来说，如果你的局域网络如同第三章的图3.1-1 所示，也就是说，你的 ppp0 可以连上 Internet ，但是内网则使用 eth0 来跟其他内部主机联机时， 那么你的 IP 设定参数：/etc/sysconfig/network-scripts/ifcfg-eth0 应该要给予一个私有 IP 以使内部的 LAN 也可以透过 eth0 来进行联机啊！所以鸟哥会这样设定：

  ```
  [root@www ~]# vim /etc/sysconfig/network-scripts/ifcfg-eth0
  DEVICE=eth0
  BOOTPROTO=none
  NM_CONTROLLED=no
  IPADDR=192.168.1.100
  NETMASK=255.255.255.0
  ONBOOT=yes
  ```

  并请记得一件事情，那就是：『千万不要有 GATEWAY 的设定！』， 因为 ppp0 拨接成功后， ISP 会主动的给予 ppp0 接口一个可以连上 Internet 的 default gateway ， 如果你又设定另一个 default gateway ，两个网关可能会造成你的网络不通喔！

- 这部主机仅有连接 ADSL 调制解调器，并没有内部网域：

  如果这部 Linux 主机是直接连接到 ADSL 调制解调器上头，并没有任何内部主机与其联机，也就是说，你的 eth0 有没有 IP 都没有关系时，那么上面的设定当中的那个『ONBOOT=yes 』直接改成『 ONBOOT=no 』就好了！那拨接不会有问题吗？ 没关系啊，因为你拨接启动 ppp0 时，系统会主动的唤醒 eth0 ，只是 eth0 不会有 IP 信息就是了。

至于其他的档案请参考 4.2.1 手动设定 IP 的联机方法来处理即可。 当然啦，拨接之前，请确认你的 ADSL 调制解调器 (小乌龟) 已经与主机联机妥当，也取得账号与密码，也安装好了 rp-pppoe ，然后就来处理吧！

1. 设定连接到 ADSL 调制解调器那张网卡 (暂订为 eth0)

   说实在的，鸟哥比较建议将内外网域分的清清楚楚比较好，所以，通常我都是主机上面接两块网络卡， 一张对内一张对外，对外的那张网卡预设是不启动的 (ONBOOT=no)。考虑到你可能仅有一张网卡，那么鸟哥也会给你建议， 直接给 eth0 一个私有 IP 接口吧！设定就如同本节稍早提到的那样啰！

2. 设定拨接的账号与密码

   好了，那么开始来设定你的账号与密码吧！这个动作只要在第一次建立账号/密码时处理即可，未来除非账号密码改变了， 否则这个动作都不需要重新处理啦！(留意一下，拨接的设定指令有改变喔！与之前的 adsl-setup 不一样啰！仔细看看！)

```
[root@www ~]# pppoe-setup
Welcome to the PPPoE client setup.  First, I will run some checks on
your system to make sure the PPPoE client is installed properly...

LOGIN NAME  (从 ISP 处取得的账号填入处)
Enter your Login Name (default root): T1234567
# 注意啊！这个账号名称是 ISP 给的，其中如果是 SeedNet ，输入如上，
# 如果是 Hinet 的话，就得要输入 username@hinet.net，后面的主机名也要写。

INTERFACE  (ADSL 调制解调器所接的网卡代号)
Enter the Ethernet interface connected to the PPPoE modem
For Solaris, this is likely to be something like /dev/hme0.
For Linux, it will be ethX, where 'X' is a number.
(default eth0): eth0

Enter the demand value (default no): no

DNS  (就填入 ISP 处取得的 DNS 号码吧)
Enter the DNS information here: 168.95.1.1
Enter the secondary DNS server address here: &lt;==若无第二部就按 enter

PASSWORD  (从 ISP 处取得的密码啊！)
Please enter your Password: &lt;==输入密码两次，屏幕不会有星号 * 喔！
Please re-enter your Password:

USERCTRL  (要不要让一般用户启动与关闭？最好是不要！)
Please enter 'yes' (three letters, lower-case.) if you want to allow
normal user to start or stop DSL connection (default yes): no

FIREWALLING  (防火墙方面，先取消，用自己未来设定的)
The firewall choices are:
0 - NONE: This script will not set any firewall rules.  You are responsible
          for ensuring the security of your machine.  You are STRONGLY
          recommended to use some kind of firewall rules.
1 - STANDALONE: Appropriate for a basic stand-alone web-surfing workstation
2 - MASQUERADE: Appropriate for a machine acting as an Internet gateway
                for a LAN
Choose a type of firewall (0-2): 0

Start this connection at boot time (要不要开机立即启动拨接程序？)
Do you want to start this connection at boot time?
Please enter no or yes (default no):yes

** Summary of what you entered **
Ethernet Interface: eth0
User name:          T1234567
Activate-on-demand: No
Primary DNS:        168.95.1.1
Firewalling:        NONE
User Control:       no
Accept these settings and adjust configuration files (y/n)? y
Adjusting /etc/sysconfig/network-scripts/ifcfg-ppp0
Adjusting /etc/resolv.conf
  (But first backing it up to /etc/resolv.conf.bak)
Adjusting /etc/ppp/chap-secrets and /etc/ppp/pap-secrets
  (But first backing it up to /etc/ppp/chap-secrets.bak)
  (But first backing it up to /etc/ppp/pap-secrets.bak)
# 上面具有特殊字体的档案主要功能是：
# ifcfg-ppp0   ：亦即是 ppp0 这个网络接口的配置文件案；
# resolv.conf  ：这个档案会被备份后，然后以刚刚我们上面输入的 DNS 数据取代；
# pap-secrets, chap-secrets：我们输入的密码就放在这里！
```

这样设定就成功啦！很简单吧！唯一需要注意的是在上面的 username 那个地方，千万注意， 因为 hinet 与 seednet 的设定是不一样的！千万小心呢！否则会无法连上线呦！此外， 由于我们在未来还会有 firewall 的建置，所以这里不太需要使用到防火墙啦！否则也可能无法连上 Internet 呦！另外，注意一下，一般拨接需要的身份认证机制透过的

是 chap 与 pap(注2)， 在 rp-pppoe 这套软件中，就将两种认证机制所需的数据通通记录下来啦！那就是 chap-secrets, pap-secrets， 你可以分别察看两个档案的内容，就知道那是啥咚咚了！

3. 透过 adsl-start, pppoe-start 或 network restart 开始拨接上网

 启动 ADSL 的方法很多，通常鸟哥都是使用 /etc/init.d/network restart 即可处理！不过，如果发生一些不明的错误， 也可以使用 pppoe-stop 关闭后再以 pppoe-start 立即启动拨接试看看。 通常比较容易出问题的地方在于硬件的联机情况，请先确认所有的硬件联机没有问题喔！ 通常，如果你使用小乌龟 (ATU-R) 时，请使用跳线连接网络卡与 ATU-R。另外一个容易出错的地方在于输入的账号与密码， 账号与密码都是你的 ISP 给你的，并且注意大小写(可以到 /etc/ppp/{chap,pap}-secrets 察看一下是否设定错误？)

4. 开始检查的步骤：

 上面的步骤搞定就可以连上 Internet 了。如果担心设定方面有问题，可以透过手动设定 IP 的那个小节的五个步骤去检查看看， 指令分别是：

 ```
 [root@www ~]# ifconfig
 [root@www ~]# route -n
 [root@www ~]# ping GW的IP
 [root@www ~]# dig www.google.com
 [root@www ~]# hostname
 ```

 比较特殊的是，因为 ADSL 拨接是透过点对点 (ppp) 协议，所谓的点对点，就是你的 ppp0 直接连接到 ISP 的某个点 (IP) ， 所以，理论上，ppp0 是个独立的 IP ，并没有子网！因此，当你察看 ppp0 的网络参数时，他会变成这样：

 ```
 [root@www ~]# ifconfig ppp0
 ppp0      Link encap:Point-to-Point Protocol
           inet addr:111.255.69.90  P-t-P:168.95.98.254  Mask:255.255.255.255
           UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1492  Metric:1
           RX packets:59 errors:0 dropped:0 overruns:0 frame:0
           TX packets:59 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:3
           RX bytes:7155 (6.9 KiB)  TX bytes:8630 (8.4 KiB)
 ```

 如上所示，那个 inet addr 就是你的 IP ，而 P-t-P 就是 Gateway 的意思啦！你也会看到， Mask 是 255.255.255.255 哩！ 没有子网呦！要仔细看清楚哩！

5. 取消拨接功能 (Option)

 如果你明明没有 ADSL 联机，但是却作了上面的动作，那么得要注意喔，因为每次重新启动网络都会花很多时间在侦测 ADSL 调制解调器上。所以啰，我们得要修改 ppp0 的配置文件才行。动作很简单，将 /etc/sysconfig/network-scripts/ifcfg-ppp0 内的 ONBOOT 改成 no， 然后进行：

```
[root@www ~]# vim /etc/sysconfig/network-scripts/ifcfg-ppp0
DEVICE=ppp0
ONBOOT=no
....(其他省略)....

[root@www ~]# chkconfig pppoe-server off
```

很快的，这样你就已经做好 ADSL 拨接上网的动作了！很快乐吧！但是不要忘记了，你的主机若还没有更新 (update) 系统，恐怕资安方面会有些问题哩！所以，赶紧往下两个章节读读去！

4.3 无线网络--以笔记本电脑为例

除了使用实体 RJ-45 线路来连接网络之外，由于现在笔记本电脑渐渐广为使用，因此在笔记本电脑上面的无线网络 (Wireless Local Area Network, WLAN) 也越来越重要啰～针对无线网络所提出的标准中，早期是 IEEE 802.11b / 802.11g 较为重要，其中 802.11g 这个标准的传输速度已经可以达到 54Mbps 的水平。不过，近期以来还有新的标准，那就是 802.11n (注3)，这个标准的理论传输速度甚至可达 300Mbps 哩！所以啰，我们也得稍微介绍一下无线网络啦！

Tips: 无线网络的机制非常多，我们现在常听到的主要有 Wi-Fi (可想成是 802.11 相关标准) 以及 WiMAX (802.16, 注4) 等， 在底下我们主要介绍的是目前使用较广泛的 Wi-Fi 相关无线网卡喔！可不要搞错啰！



4.3.1 无线网络所需要的硬件： AP、无线网卡

我们知道在 RJ-45 的以太网络联机环境中，以 switch/hub 以及网络卡与网络线最重要，该架构中主要以 switch/hub 串接所有的网络设备。那么在无线网络中，当然也需要一个接收讯号的装置，那就是无线基地台 (Wireless Access Point, 简称 AP) 了！另一个装置当然就是安装在计算机主机上面的无线网卡啰！

其实无线基地台本身就是个 IP 分享器了，他本身会有两个接口，一个可以与外部的 IP 做沟通，另外一个则是作为 LAN 内部其他主机的 GATEWAY 啰！那其他主机上面只要安装了无线网卡，并且顺利的连上 AP 后，自然就可以透过 AP 来连上 Internet 啦！整个传输的情况可以用下图来示意：

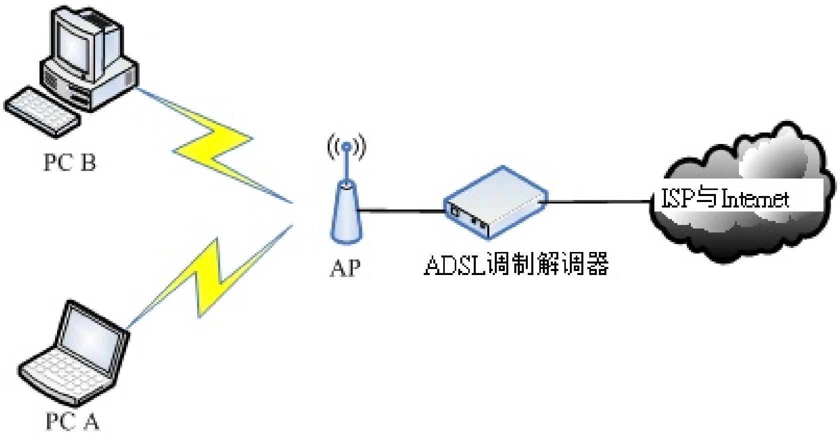


图 4.3-1、无线网络的联机图标

在上图中，我们假设 PC A 与 PC B 这两部主机都有安装无线网卡，因此他们可以扫瞄到局域网络内的 AP 存在，所以可以透过这个 AP 来连上 Internet 啊。在不考虑内部 LAN 联机的情况下，AP 如何连上 Internet 呢？虽然每部 AP 的控制接口都不相同，不过绝大部分的 AP 都是提供 Web 接口来设定的，因此你可以参考每部 AP 的说明书来进行设定，在这里鸟哥就不多说了。

鸟哥就以手边有的设备来说明这个项目，使用的设备如下：

- AP ： TP-Link (TL-WR941ND)
- USB 的无线网卡： D-Link (DWA-140)，使用 RT3070sta 驱动程序

比较凄惨的是，CentOS 6.x 预设不支持 DWA-140 这个 USB 的无线网卡，因此原本我们还得要自行手动下载 USB 无线网卡的驱动程序才行！更怪的是，我们的核心侦测到的模块是 rt2870sta，但实际上该硬件使用的是 rt3070sta模块... 为了这个，搞了鸟哥两、三天的时间去解决问题...还好，由世界上热心的网友回报支持 Linux 的无线网卡网站说明，发现这只 USB 是支援 Linux 的喔！如下网址所示。而且，已经有公司将这个网卡编译成 CentOS 6.x 可以使用的 RPM 档案啰！相关网址如下：

- 网友们热心提供：http://linux-wless.passys.nl/query_part.php?brandname=D-Link
- 帮我们打包成 RPM 的公司：http://rpm.pbone.net/index.php3
- Ralink 官网的下载处：http://www.ralinktech.com/support.php?s=2

鸟哥最终由上面第二个网址下载的两个档案是：kmod-rt3070sta-2.5.0.1-2.el6.elrepo.x86_64.rpm, rt2870-firmware-22-1.el6.elrepo.noarch.rpm。鸟哥将他放置于 /root 底下，等一下再来安装。

Tips: 这张 USB 无线网卡让鸟哥搞到一个头两个大！基本上，Linux 核心预设不支持的设备，建议不要购买啦！ 否则很难处理！鸟哥觉得这个 DWA-140 感觉就是张恶魔卡～好怪～好难搞...



4.3.2 关于 AP 的设定：网络安全方面

如果你留心一下图 4.3-1 ，那么就可以发现一件事情，那就是：『如果 AP 不设定任何联机限制，那任何拥有无线网卡的主机都可以透过这个 AP 连接上你的 LAN』， 要知道，通常我们都会认为 LAN 是信任网域，所以内部是没有防火墙的，亦即是不设防的状态，呵呵！ 如果刚好有人拿着笔记本电脑经过你的 AP 可以接收讯号的范围，那么他就可以轻易的透过你的 AP 连接上你的 LAN ，并且可以透过你的 AP 连上 Internet ，如果他刚好是个喜欢搞破坏的 cracker ， 哈哈！那么当他使用你的 AP 去攻击别人时，最后被发现的跳板是谁？当然是你的 AP！ 那是谁会吃上官司？够清楚了吧？而且你内部主机的数据也很有可能被窃取啊！

所以啦，『无线网络的安全性一定是具有很大的漏洞的』，没办法， 因为无线网络的传输并不是透过实体的网络线，而是透过无线讯号，实体网络线很好控制，无线讯号你如何侦测啊？对吧！ 因此，请你务必在你的 AP 上面进行好联机的限制设定，一般可以这样做限制的：

- 在 AP 上面使用网卡卡号 (MAC) 来作为是否可以存取 AP 的限制：

  如此一来，就只有你允许的网络卡才能够存取你的 AP ，当然会安全不少。不过这个方法有个问题， 那就是当有其他主机想要透过这个 AP 联机时，你就得要手动的登入 AP 去进行 MAC 的设定， 在经常有变动性装置的环境中 (例如公司行号或学校)，这个方法比较麻烦～

- 设定你的 AP 联机加密机制与密钥：

  另一个比较可行的办法就是设定联机时所需要的验证密钥！这个密钥不但可以在网络联机的数据当中加密，使得即使你的数据被窃听， 对方也是仅能得到一堆乱码，同时由于 client 端也需要知道密钥并且在联机阶段输入密钥， 因此也可以被用来限制可联机的用户啊！

当然，上面两种方法你可以同时设定，亦即不但需要联机的密钥，而且在 AP 处也设定能够存取的 MAC 网卡， 嘿嘿！这样一来，就更安全的多了(注5)。底下让我们来介绍一下 AP 里面经常要了解的数据， 那就是 ESSID/SSID 啰！

- 关于 ESSID/SSID ：

想一想，如果你有两部 AP 在同一个局域网络内，那么请问一下，当你的无线网卡在上网时，他会透过哪一个 AP 联机出去呢？很困扰，对吧！其实每部 AP 都会有一个联机的名字，那就是 SSID 或 ESSID，这个 SSID 可以提供给 client 端， 当 client 端需要进行无线联机时，他

必须要说明他要利用哪一部 AP ，那个 ESSID 就是那时需要输入的数据了！在鸟哥的案例当中，我将我的 AP 设定为 vbird_tsai 这个名字， 并且给予一个密钥密码，设定的方法如同下图所示：



图 4.3-2、无线网络 AP 的 SSID 设定项目

如上图，在登入了 AP 的设定项目后，依序 (1)先选择无线网络里面的『无线网络设定』，然后在右边的窗口当中 (2)填写正确的 SSID 号码，然后按下 (3)储存即可。之后就是密码项目啦！密码项目的设定画面如下：

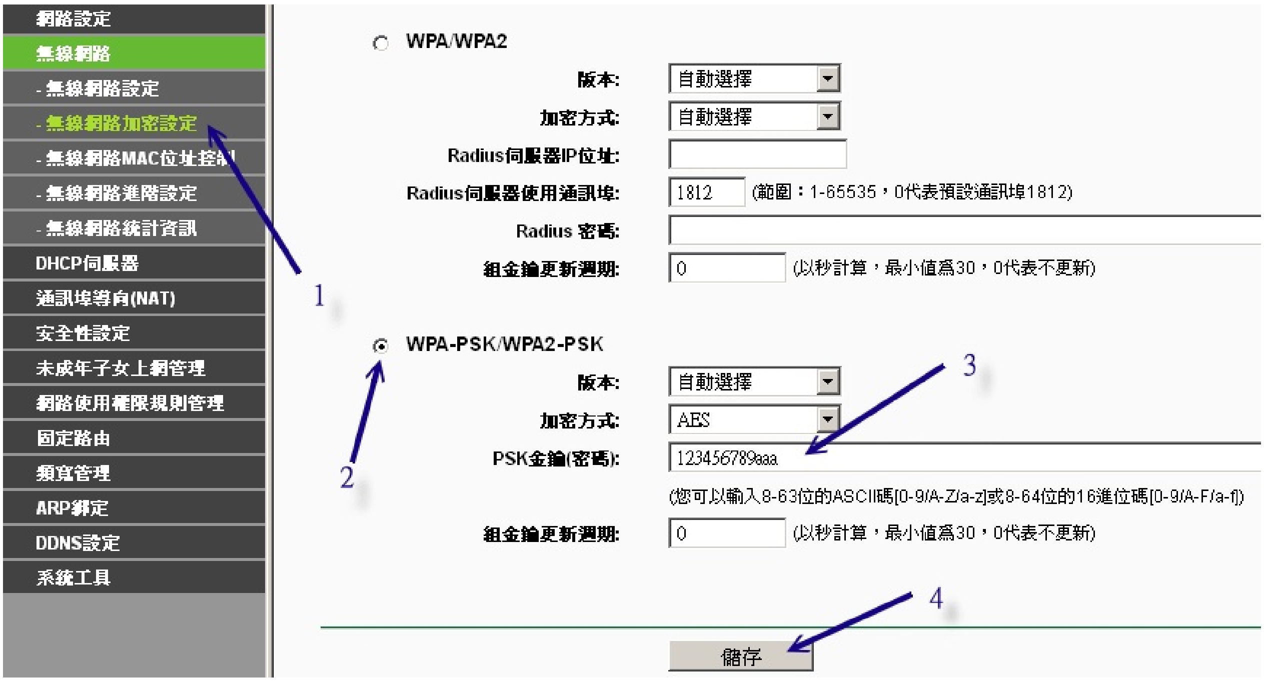


图 4.3-3、无线网络 AP 的密钥设定项目

我们先选择 (1)无线网络加密设定，然后在右边窗口 (2)点选 WPA-PSK/WPA2-PSK 的加密方式，然后 (3)输入加密的密钥长度，鸟哥这里填写的算是简单到爆炸的密码，小朋友不要学喔！填完后按下储存即可。 这个时候我们就会有底下两个数据：

- SSID： vbird_tsai
- 密钥密码： 123456789aaa

这仅是个范例说明！ AP 设定就到此为止，如果您的设定有不同的地方，请自行查询您 AP 的操作手册呦！

4.3.3 利用无线网卡开始联机

无线网卡有很多模式，鸟哥选择的是 USB 无线网卡，所以想要知道有没有捉到这张网卡，就得要使用 lsusb 来检查， 如果核心预设不支持，还得要自行编译驱动程序才行！如前所述，我们的驱动程序已经捉在 /root 底下了！

1. 检查无线网卡的硬件装置：

   使用 USB 无线网卡的检查方式如下：

   ```
   [root@www ~]# lsusb
   Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
   Bus 001 Device 003: ID 07d1:3c0a D-Link System DWA-140 RangeBooster
   N Adapter(rev.B2) [Ralink RT2870]
   Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
   # 是有捉到的！只是，有加载吗？不知道呢！继续往下检查看看！
   ```

2. 察看模块与相对应的网卡代号：(modinfo 与 iwconfig)

知道核心侦测到这张网卡，但是能不能正确的加载模块呢？来瞧瞧：

```
[root@www ~]# iwconfig
lo        no wireless extensions.
eth0      no wireless extensions.
# 要出现名为 wlan0 之类的网卡才是有捉到喔！所以没有加载正确模块啦！
```

因为没有加载正确的驱动程序，现在让我们来安装刚刚下载的 RPM 驱动程序吧！请先将 USB 拔出来， 然后再安装 RPM 档案。安装的方法不要说你忘记了！

```
[root@www ~]# rpm -ivh kmod-rt3070sta* rt2870-firmware*
# 这个动作会进行很久，似乎程序在侦测硬件的样子！
# 这个咚咚做完之后，请将 USB 网卡插入 USB 插槽吧！

[root@www ~]# iwconfig
lo        no wireless extensions.
eth0      no wireless extensions.
ra0       Ralink STA
```

这个 iwconfig 是用在作为无线网络设定之用的一个指令，与 ifconfig 类似！不过，当我们使用 iwconfig 时，如果有发现上述的特殊字体，那就代表该网络接口使用的是无线网卡的意思啊！虽然有时你会看到无线网卡为 wlan0 之类的代号，不过这张网卡却使用 ra0 作为代号，挺有趣的！

3. 利用 iwlist 侦测 AP ：

好了，接下来要干嘛？当然是看看我们的无线网卡是否能够找到 AP 啊！所以，首先我们要启动无线网卡，就利用 ifconfig 即可：

```
[root@www ~]# ifconfig ra0 up
```

启动网卡后才能以这个网卡来搜寻整个区域内的无线基地台啊！接下来，直接使用 iwlist 来使用这个无线网卡搜寻看看吧！

```
[root@www ~]# iwlist ra0 scan
ra0       Scan completed :
          Cell 01 - Address: 74:EA:3A:C9:EE:1A
                    Protocol:802.11b/g/n
                    ESSID:"vbird_tsai"
                    Mode:Managed
                    Frequency:2.437 GHz (Channel 6)
                    Quality=100/100  Signal level=-45 dBm  Noise level=-92 dBm
                    Encryption key:on
                    Bit Rates:54 Mb/s
                    IE: WPA Version 1
                        Group Cipher : CCMP
                        Pairwise Ciphers (1) : CCMP
                        Authentication Suites (1) : PSK
                    IE: IEEE 802.11i/WPA2 Version 1
                        Group Cipher : CCMP
                        Pairwise Ciphers (1) : CCMP
                        Authentication Suites (1) : PSK
....(底下省略)....
```

从上面可以看到 (1)这个无线 AP 的协议，并且也能够知道 (2)ESSID 的名号是没错的！当然啦，(3)连加密的机制是 WPA2-PSK 也是能够得知的！这与前一小节的 AP 设定是相符合的！(4)使用的无线频道是 6 号，接下来呢？就得要去修改配置文件，这部份很麻烦，请参考如下的网页来设定：

- https://wiki.archlinux.org/index.php/Rt2870

```
[root@www ~]# ifconfig ra0 down && rmmod rt3070sta
[root@www ~]# vim /etc/Wireless/RT2870STA/RT2870STA.dat
Default
CountryRegion=5
CountryRegionABand=7
CountryCode=TW            &lt;==台湾的国码代号！
ChannelGeography=1
SSID=vbird_tsai           &lt;==你的 AP 的 ESSID 喔！
NetworkType=Infra
WirelessMode=9            &lt;==与无线 AP 支持的协议有关！参考上述网址说明
Channel=6                 &lt;==与 CountryRegion 及侦测到的频道有关的设定！
....(中间省略)....
AuthMode=WPAPSK           &lt;==我们的 AP 提供的认证模式
EncrypType=AES            &lt;==传送认证码的加密机制啊！
WPAPSK="123456780aaa"     &lt;==密钥密码！最好用双引号括起来较佳！
....(底下省略)....
# 鸟哥实际有修改的，就是上面有特别说明的地方，其余的地方都保留默认值即可。
# 更奇怪的是，每次 ifconfig ra0 down 后，这个档案会莫名其妙的修改掉 @_@

[root@www ~]# modprobe rt3070sta && ifconfig ra0 up
[root@www ~]# iwconfig ra0
ra0       Ralink STA  ESSID:"vbird_tsai"  Nickname:"RT2870STA"
          Mode:Auto  Frequency=2.437 GHz  Access Point: 74:EA:3A:C9:EE:1A
          Bit Rate=1 Mb/s
          RTS thr:off   Fragment thr:off
          Encryption key:off
          Link Quality=100/100  Signal level:-37 dBm  Noise level:-37 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

如果顺利出现上面的数据，那就表示你的无线网卡已经与 AP 接上线了～再来则是设定网络卡的配置文件啰！^_^

4. 设定网络卡配置文件 (ifcfg-ethn)

因为我们的网络卡使用的代号是 ra0，所以也是需要在 /etc/sysconfig/network-scripts 设定好相对应的档案才行啊！而由于我们的这块卡其实是无线网卡， 所以很多设定值都与原本的以太网络卡不同，详细的各项变量设定你可以自行参考一下底下的档案：

- /etc/sysconfig/network-scripts/ifup-wireless 至于我的网络卡设定是这样的：

```
[root@www ~]# cd /etc/sysconfig/network-scripts
[root@www network-scripts]# vim ifcfg-ra0
DEVICE=ra0
BOOTPROTO=dhcp
ONBOOT=no    &lt;== 若需要每次都自动启动，改成 yes 即可！
ESSID=vbird_tsai
RATE=54M     &lt;== 可以严格指定传输的速率，要与上面 iwconfig 相同，单位 b/s
```

要注意的是那个 ONBOOT=no 的设定，如果你想要每次开机时无线，网卡都会自动启动， 那就将他设定为 yes 吧！否则就设定为 no 啰！要启动再以 ifup ra0 来启动即可！呼呼！到此为止，你的无线网卡已经可以顺利的给他启动了喔！很快乐吧！ ^_^

Tips: 其实上面那个配置文件的内容都是在规划出 iwconfig 的参数而已，所以你除了可以查阅 ifup-wireless 的内容外，可以 man iwconfig ，会知道的更详细喔！而最重要的参数当然就是 ESSID 及 KEY 啰！ ^_^



1. 启动与观察无线网卡

   要启动就用 ifup wlan0 来启动，很简单啦！要观察就用 iwconfig 及 ifconfig 分别观察，底下你自己瞧瞧就好啊！ ^_^

```
[root@www ~]# ifup ra0
Determining IP information for ra0... done.
```

整个流程就是这么简单喔！一般来说，目前比较常见的笔记本电脑内建的 Intel 无线网络模块 (Centrino) 适用于 Linux 的 ipw2200/ipw21000 模块，所以设定上也是很快！因为 CentOS 6.x 预设就有支持，你不必重新安装无线网卡驱动程序！ 那直接透过上述的方式来处理你的无线网络即可！很快速又方便吧！本章结尾的参考数据处， 鸟哥还是列出许多与无线网卡有关的连结，你可以自行前往查阅与你的无线网卡有关的信息喔(注6)！ ^_^

4.4 常见问题说明

其实这个小节也很重要的！因为可以让你在念完理论后，了解一下如何利用那些概念来查询你的网络设定问题喔！ 底下我们就针对几个常见的问题来说说看吧！

4.4.1 内部网域使用某些联机服务(如 FTP, POP3)所遇到的联机延迟问题

你或许曾经听过这样的问题：『我在我的内部区域网域内有几部计算机， 这几部计算机明明都是在同一个网域之内，而且系统通通没有问题，为什么我使用 pop3 或者是 ftp 连上我的 Linux 主机会停顿好久才连上？但是连上之后，速度就又恢复正常！』

由于网络在联机时，两部主机之间会互相询问对方的主机名配合的 IP ，以确认对方的身份。在目前的因特网上面，我们大多使用 Domain Name System (DNS) 系统做为主机名与 IP 对应的查询，那就是我们在上面提到的 /etc/resolv.conf 档案内设定的 IP 由来， 如果没有指定正确的 DNS IP 的话，那么我们就无法查询到主机名与 IP 的对应了。

公开的因特网可以这样设定，但是如果是我们内部网域的私有 IP 主机呢？ 因为是私有 IP 的主机，所以当然无法使用 /etc/resolv.conf 的设定来查询到这部主机的名称啊！ 那怎么办？要知道，如果两部主机之间无法查询到正确的主机名与 IP 的对应， 那么将『可能』发生持续查询主机名对应的动作，这个动作一般需要持续 30-60 秒，因此，你的该次联机将会持续检查主机名 30 秒钟，也就会造成奇怪的 delay 的情况。

这个问题最常发生在内部的 LAN ，例如使用 192.168.1.1 的主机联机到 192.168.1.2 的主机。 这个问题虽然可以透过修改软件的设定来略过主机名的检查，但是绝大多数的软件都是默认启用这个机制的， 因此，内部主机『老是联机时期很慢，联机成功后速度就会恢复正常』 时，通常就是这个问题啦！尤其是在 FTP 及 POP3 等网络联机软件上最常见。

那么如何避过这个情况？最简单的方法就是『给予内部的主机每部主机一个名称与 IP 的对应』即可。举例来说，我们知道每部主机都有一个主机名为 localhost ，对应到 127.0.0.1 ，为什么呢？因为这个 127.0.0.1 与 localhost 的对应就被写到 /etc/hosts 内嘛！ 当我们需要主机名与 IP 的对应时，系统就会先到 /etc/hosts 找寻对应的设定值， 如果找不到，才会使用 /etc/resolv.conf 的设定去因特网找。这样说，你明白了吧？ 也就是说，只要修改了 /etc/hosts，加入每部主机与 IP 的对应， 就能够加快主机名的检查啰！

了解了吗？所以说，你就要将你的 私有 IP 的计算机与计算机名称写入你的 /etc/hosts 当中了！这也是为啥我们在主机名设定的地方， 特别强调第五个检查步骤的缘故。我们来看一看 /etc/hosts 原本的设定内容吧！

```
[root@www ~]# cat /etc/hosts
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1               localhost.localdomain   localhost
# 主机的 IP              主机的名称               主机的别名
```

在上面的情况中很容易就发现了设定的方法了吧！很简单吧！没错！那就是 IP 对应主机名啦！ 那么现在知道为什么我们给他 ping localhost 的时候，地址会写出 127.0.0.1 了吧！ 那就是写在这个档案中的啦！而且 localhost 那一行不能拿掉呦！ 否则系统的某些服务可能就会无法被启动！好了！那么将我局域网络内的所有的计算机 IP 都给他写进去！并且，每一部给他取一个你喜欢的名字， 即使与 client 的计算机名称设定不同也没关系啦！ 以鸟哥为例，如果我还额外加设了 DHCP 的时候，那么我就干脆将所有的 C Class 的所有网段全部给他写入 /etc/hosts 当中，有点像底下这样：

```
[root@www ~]# vim /etc/hosts
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1               localhost.localdomain   localhost
192.168.1.1    +   linux001
192.168.1.2    +   linux002
192.168.1.3    +   linux003
.........
.........
192.168.1.254  linux254
```

如此一来，不论我哪一部计算机连上来，不论是在同一个网段的哪一个 IP ， 我都可以很快速的追查到！嘿嘿！那么区内网络互连的时候，就不会多等个好几时秒钟啰！

4.4.2 网址列无法解析问题

很多朋友常问的一个问题『咦！我可以拨接上网了，也可以 ping 到奇摩雅虎的 IP ，但为何就是无法直接以网址连上 Internet 呢！』嘿！被气死！ 前面不是一直强调那个 DNS 解析的问题吗？对啦！就是名称解析不对啦！赶快改一下 /etc/resolv.conf 这个档案吧！改成上层 ISP 给你的 DNS 主机的 IP 就可以啦！例如 Hinet 的 168.95.1.1 及 Seednet 的 139.175.10.20 啰！例如底下的范例(这个范例就可以照抄了！ ^_^)：

```
[root@www ~]# vi /etc/resolv.conf
nameserver 168.95.1.1
nameserver 139.175.10.20
```

朋友们常常会在这个地方写错，因为很多书上都说这里要设定成为 NAT 主机的 IP ， 那根本就是不对的！你应该要将所有管理的计算机内，关于 DNS 的设定都直接使用上面的设定值即可！ 除非你的上层环境有使用防火墙，那才另外考虑！

4.4.3 预设路由的问题

记得我们在前两章提到的网络基础当中，不是讲了很多预设路由 (default gateway) 相关的说明吗？ 预设路由通常仅有一个，用来做为同一网域的其他主机传递非本网域的封包网关。 但我们也知道在每个网络配置文件案 (/etc/sysconfig/network-scripts/ifcfg-ethx) 内部都可以指定『 GATEWAY 』这个参数，若这个参数重复设定的话，那可就麻烦啦！

举例来说，你的 ifcfg-eth0 用来做为内部网域的沟通，所以你在该档案内设定 GATEWAY 为你自己的 IP ， 但是该主机为使用 ADSL 拨接，所以当拨接成功后会产生一个 ppp0 的接口，这个 ppp0 接口也有自己的 default gateway ，好了，那么当你要将封包传送到 Yahoo 这个非为本网域的主机时， 这个封包是要传到 eth0 还是 ppp0 呢？因为两个都有 default gateway 啊！

没错！很多朋友就是这里搞不懂啦！常常会错乱～所以，请注意， 你的 default gateway 应该只能有一个， 如果是拨接，请不要在 ifcfg-eth0 当中指定 GATEWAY或 GATEWAYDEV 等变量，重要重要！

更多的网络除错请参考后续第六章 Linux 网络侦错的说明。

4.5 重点回顾

- Linux 以太网络卡的默认代号为 eth0, eth1 等等, 无线网卡则为 wlan0, ra0 等等；
- 若需要自行编译网卡驱动程序时，则你必须要先安装 gcc, make, kernel-header 等软件。
- 内部网域的私有 IP 之主机的『 IP 与主机名的对应』，最好还是写入 /etc/hosts ， 可以克服很多软件的 IP 反查所花费的等待时间。
- IP 参数设定在 /etc/sysconfig/network-scripts/ifcfg-eth0 当中，主机名设定在 /etc/sysconfig/network 当中，DNS 设定在 /etc/resolv.conf 当中，主机名与 IP 的对应设定在 /etc/hosts；
- 在 GATEWAY 这个参数的设定上面，务必检查妥当，仅设定一个 GATEWAY 即可。
- 可以使用 /etc/init.d/network restart 来重新启动整个系统的网络接口。
- 若使用 DHCP 协议时，则请将 GATEWAY 取消设定，避免重复出现多个 default gateway ，反而造成无法联机的状况。
- ADSL 拨接后可以产生一个新的实体接口，名称为 ppp0
- 无线网卡与无线基地台之间的联机由于是透过无线接口，所以需要特别注意网络安全；
- 常见的无线基地台(AP)的联机防护，主要利用控制登入者的 MAC 或者是加上联机加密机制的密钥等方法；
- 设定网络卡可以使用 ifconfig 这个指令，而设定无线网卡则需要 iwconfig ，至于扫瞄基地台， 可以使用 iwlist 这个指令。

4.6 本章习题

- 我要如何确定我在 Linux 系统上面的网络卡已经被 Linux 捉到并且驱动了？网络卡能不能被捉到可以使用『 dmesg|grep eth 』来判断，有没有驱动则可以使用 lsmod 看看模块有没有加载核心！最后，以 ifconfig eth0 192.168.0.10 测试看看！
- 假设我的网络参数为：IP 192.168.100.100, Netmask 255.255.255.0, 请问我要如何在 Linux 上面设定好这些网络参数 (未提及的网络参数请自行定义！)？请使用手动与档案设定方法分别说明。手动设定为：『 ifconfig eth0 192.168.100.100 netmask 255.255.255.0 up 』 档案设定为：vi /etc/sysconfig/network-scripts/ifcfg-eth0 ，内容为： DEVICE=eth0 ONBOOT=yes BOOTPROTO=static IPADDR=192.168.100.100 NETMASK=255.255.255.0 NETWORK=192.168.100.0 BROADCAST=192.168.100.255 要启动则使用 ifup eth0 即可！
- 我要将我的 Linux 主机名改名字，步骤应该如何(更改那个档案？如何启用？)？Linux 主机名在 /etc/sysconfig/network 这个档案里面的『HOSTNAME=主机名』来设定，先以 vi 来修改，改完后可以使用 /etc/init.d/network restart 不过建议直接 reboot 启动主机名！
- /etc/resolv.conf 与 /etc/hosts 的功能为何？以主机名寻找 IP 的方法， /etc/resolv.conf 内填写 DNS 主机名，至于 /etc/hosts 则直接填写主机名对应的 IP 即可！ 其中 /etc/hosts 对于内部私有 IP 的主机名查询非常有帮助！
- 我使用 ADSL 拨接连上 Internet ，请问拨接成功之后，我的 Linux 上面会有几个网络接口 (假设我只有一个网络卡)？因为拨接是使用 PPP (点对点)协议，所以拨接成功后会多出一个 ppp0 的接口，此外，系统原本即有 eth0 及 lo 这两个界面，所以共有三个界面。
- 一般来说，如果我拨接成功，也取得了 ppp0 这个接口，但是却无法对外联机成功， 你认为应该是哪里出了问题？该如何解决？因为拨接成功了，表示物理对外联机没有问题，那么可能的问题应该是发生在 Gateway 上面了！确认的方法请使用 route -n 查阅路由信息，然后修订 /etc/sysconfig/network-scripts/ifcfg-eth0 吧！
- 如果你的局域网络环境内有可以控管的无线 AP 时，请自行查出如何以 MAC 的方式管理可登入的用户， 并将你的无线 AP 做好联机加密的密钥设定。请自行测试！谢谢！
- 如果一部主机上面插了两张相同芯片的网络卡，代表两者使用的模块为同一个，此时可能会造成网卡代号的误判；请问你如何克服这个问题？让网卡代号不会变动？以现在的方法来讲，其实我们可以透过指定 Hardware Address(硬件地址，通称为 MAC) 来指定网卡代号与 MAC 的对应。 这个设定值可以在 ifcfg-ethx 里面以 HWADDR 这个设定项目来指定的。
- 如何在 Linux 上面的文字接口搜寻你所在区域的无线 AP ？透过直接使用『 iwlist scan 』这个指令来指定某个无线网卡的搜寻！ 然后再以 iwconfig 来进行网卡的设定即可！
- 请依序说明：如果你想要新增一块新的网络卡在你的主机上，并给予一个固定的私有 IP ，应如何进行？
    - 先关掉主机的 power ，然后拆掉机壳，装上网络卡；
    - 开机完成后，以 dmesg | grep eth 查询是否可捉到该网络卡，若无法捉到，请编译

模块，若可捉到，找出网卡代号， 并且将该模块与网卡代号写入 /etc/modprobe.conf 当中，以利未来开机时可自动达成对应；
    - 利用『 ifconfig "网卡代号" 』来查询 MAC 为何？
    - 开始在 /etc/sysconfig/network-scripts 内建立 ifcfg-"网卡代号" 档案，同时给予 HWADDR 的对应；
    - 启动 /etc/init.d/network restart 测试是否能成功！
- 如果你想要登入某个区域的无线 AP ，你应该向该处所至少申请哪些数据？无线网络的技术相当多且复杂，所以需要取得的参数都不尽相同。不过，至少你还是得要取得 ESSID 以及 KEY 密码，这样才能够联机登入该 AP 当中。

4.7 参考数据与延伸阅读

- 注1：rp-pppoe 官方网站：http://www.roaringpenguin.com/pppoe/ rp-pppoe 的安装方法：http://linux.vbird.org/linux_server/0130internet_connect/0130internet_connect.php#connect_adsl
- 注2：相关的认证说明： chap: http://en.wikipedia.org/wiki/Challenge-handshake_authentication_protocol pap: http://en.wikipedia.org/wiki/Password_authentication_protocol
- 注3：802.11n 在维基百科的说明：http://en.wikipedia.org/wiki/IEEE_802.11n-2009
- 注4：Wi-Fi http://zh.wikipedia.org/zh-tw/WiFi WiMAX http://zh.wikipedia.org/wiki/WiMAX?variant=zh-tw
- 注5：无线网络安全白皮书：http://www.cert.org.tw/document/docfile/Wireless_Security.pdf
- 注6：Intel Centrino 的无线网卡相关模块信息： http://ipw2100.sourceforge.net/, http://ipw2200.sourceforge.net/ HP 的许多无线网络的计划链接：http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/

2002/07/24：首次释出日期 2003/08/18：重新校正，并且修正部分书写方式 2003/08/20：新增课后练习部分 2003/09/19：加入 前往参考用解答 2006/07/17：将原本旧文章移动到 此处 2010/08/21：将原本基于 CentOS 4.x 的文章移动到 此处 2010/08/27：由于目前的硬件环境不同了，所以修改了无线网络的处理方式！ 2010/08/28：终于修改完毕！在无线网卡的地方耽误太多时间了～没有设备啊！ 2011/07/15：将基于 CentOS 5.x 的文章移动到此处 2011/07/20：有够难处理的一篇文章！尤其是无线网络的环境重现部分，好麻烦～

第五章、 Linux 常用网络指令

最近更新日期：2011/07/18

Linux 的网络功能相当的强悍，一时之间我们也无法完全的介绍所有的网络指令，这个章节主要的目的在介绍一些常见的网络指令而已。 至于每个指令的详细用途将在后续服务器架设时，依照指令的相关性来进行说明。当然，在这个章节的主要目的是在于将所有的指令汇整在一起， 比较容易了解啦！这一章还有个相当重要的重点，那就是封包撷取的指令。若不熟悉也没关系，先放着，全部读完后再回来这一章仔细练习啊！

5.1 网络参数设定使用的指令

任何时刻如果你想要做好你的网络参数设定，包括 IP 参数、路由参数与无线网络等等，就得要了解底下这些相关的指令才行！其中以 ifconfig 及 route 这两支指令算是较重要的喔！ ^_^！当然，比较新鲜的作法，可以使用 ip 这个汇整的指令来设定 IP 参数啦！

- ifconfig ：查询、设定网络卡与 IP 网域等相关参数；
- ifup, ifdown：这两个档案是 script，透过更简单的方式来启动网络接口；
- route ：查询、设定路由表 (route table)
- ip ：复合式的指令， 可以直接修改上述提到的功能；

5.1.1 手动/自动设定与启动/关闭 IP 参数： ifconfig, ifup, ifdown

这三个指令的用途都是在启动网络接口，不过， ifup 与 ifdown 仅能就 /etc/sysconfig/network-scripts 内的 ifcfg-ethX (X 为数字) 进行启动或关闭的动作，并不能直接修改网络参数，除非手动调整 ifcfg-ethX 档案才行。至于 ifconfig 则可以直接手动给予某个接口 IP 或调整其网络参数！底下我们就分别来谈一谈！

- ifconfig

ifconfig 主要是可以手动的启动、观察与修改网络接口的相关参数，可以修改的参数很多啊，包括 IP 参数以及 MTU 等等都可以修改，他的语法如下：

```
[root@www ~]# ifconfig {interface} {up&#124;down}  &lt;== 观察与启动接口
[root@www ~]# ifconfig interface {options}    &lt;== 设定与修改接口
选项与参数：
interface：网络卡接口代号，包括 eth0, eth1, ppp0 等等
options  ：可以接的参数，包括如下：
    up, down ：启动 (up) 或关闭 (down) 该网络接口(不涉及任何参数)
    mtu      ：可以设定不同的 MTU 数值，例如 mtu 1500 (单位为 byte)
    netmask  ：就是子屏蔽网络；
    broadcast：就是广播地址啊！

# 范例一：观察所有的网络接口(直接输入 ifconfig)
[root@www ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:71:85:BD
          inet addr:192.168.1.100  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe71:85bd/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2555 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:239892 (234.2 KiB)  TX bytes:11153 (10.8 KiB)
```

一般来说，直接输入 ifconfig 就会列出目前已经被启动的卡，不论这个卡是否有给予 IP，都会被显示出来。而如果是输入 ifconfig eth0，则仅会秀出这张接口的相关数据， 而不管该接口是否有启动。所以如果你想要知道某张网络卡的 Hardware Address，直接输入『 ifconfig "网络接口代号" 』即可喔！ ^_^！至于上表出现的各项数据是这样的(数据排列由上而下、由左而右)：

- eth0：就是网络卡的代号，也有 lo 这个 loopback ；
- HWaddr：就是网络卡的硬件地址，俗称的 MAC 是也；
- inet addr：IPv4 的 IP 地址，后续的 Bcast, Mask 分别代表的是 Broadcast 与 netmask 喔！
- inet6 addr：是 IPv6 的版本的 IP ，我们没有使用，所以略过；
- MTU：就是第二章谈到的 MTU 啊！
- RX：那一行代表的是网络由启动到目前为止的封包接收情况， packets 代表封包数、errors 代表封包发生错误的数量、 dropped 代表封包由于有问题而遭丢弃的数量等等
- TX：与 RX 相反，为网络由启动到目前为止的传送情况；
- collisions：代表封包碰撞的情况，如果发生太多次， 表示你的网络状况不太好；
- txqueuelen：代表用来传输数据的缓冲区的储存长度；
- RX bytes, TX bytes：总接收、发送字节总量

透过观察上述的资料，大致上可以了解到你的网络情况，尤其是那个 RX, TX 内的 error 数量， 以及是否发生严重的 collision 情况，都是需要注意的喔！ ^_^

```
# 范例二：暂时修改网络接口，给予 eth0 一个 192.168.100.100/24 的参数
[root@www ~]# ifconfig eth0 192.168.100.100
# 如果不加任何其他参数，则系统会依照该 IP 所在的 class 范围，自动的计算出
# netmask 以及 network, broadcast 等 IP 参数，若想改其他参数则：

[root@www ~]# ifconfig eth0 192.168.100.100 \
&gt; netmask 255.255.255.128 mtu 8000
# 设定不同参数的网络接口，同时设定 MTU 的数值！

[root@www ~]# ifconfig eth0 mtu 9000
# 仅修改该接口的 MTU 数值，其他的保持不动！

[root@www ~]# ifconfig eth0:0 192.168.50.50
# 仔细看那个界面是 eth0:0 喔！那就是在该实体网卡上，再仿真一个网络接口，
# 亦即是在一张网络卡上面设定多个 IP 的意思啦！

[root@www ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:71:85:BD
          inet addr:192.168.100.100  Bcast:192.168.100.127  Mask:255.255.255.128
          inet6 addr: fe80::a00:27ff:fe71:85bd/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:9000  Metric:1
          RX packets:2555 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:239892 (234.2 KiB)  TX bytes:11153 (10.8 KiB)

eth0:0    Link encap:Ethernet  HWaddr 08:00:27:71:85:BD
          inet addr:192.168.50.50  Bcast:192.168.50.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:9000  Metric:1
# 仔细看，是否与硬件有关的信息都相同啊！没错！因为是同一张网卡嘛！
# 那如果想要将刚刚建立的那张 eth0:0 关闭就好，不影响原有的 eth0 呢？

[root@www ~]# ifconfig eth0:0 down
# 关掉 eth0:0 这个界面。那如果想用默认值启动 eth1：『ifconfig eth1 up』即可达成

# 范例三：将手动的处理全部取消，使用原有的设定值重建网络参数：
[root@www ~]# /etc/init.d/network restart
# 刚刚设定的数据全部失效，会以 ifcfg-ethX 的设定为主！
```

使用 ifconfig 可以暂时手动来设定或修改某个适配卡的相关功能，并且也可以透过 eth0:0 这种虚拟的网络接口来设定好一张网络卡上面的多个 IP 喔！手动的方式真是简单啊！并且设定错误也不打紧，因为我们可以利用 /etc/init.d/network restart 来重新启动整个网络接口，那么之前手动的设定数据会全部都失效喔！另外， 要启动某个网络接口，但又不让他具有 IP 参数时，直接给他 ifconfig eth0 up 即可！ 这个动作经常在无线网卡当中会进行，因为我们必须要启动无线网卡让他去侦测 AP 存在与否啊！

- ifup, ifdown

实时的手动修改一些网络接口参数，可以利用 ifconfig 来达成，如果是要直接以配置文件，亦即是在 /etc/sysconfig/network-scripts 里面的 ifcfg-ethx 等档案的设定参数来启动的话， 那就得要透过 ifdown 或 ifup 来达成了。

```
[root@www ~]# ifup   {interface}
[root@www ~]# ifdown {interface}

[root@www ~]# ifup eth0
```

ifup 与 ifdown 真是太简单了！这两支程序其实是 script 而已，他会直接到 /etc/sysconfig/network-scripts 目录下搜寻对应的配置文件，例如 ifup eth0 时，他会找出 ifcfg-eth0 这个档案的内容，然后来加以设定。 关于 ifcfg-eth0 的设定则请参考第四章的说明。

不过，由于这两支程序主要是搜寻配置文件 (ifcfg-ethx) 来进行启动与关闭的， 所以在使用前请确定 ifcfg-ethx 是否真的存在于正确的目录内，否则会启动失败喔！ 另外，如果以 ifconfig eth0 .... 来设定或者是修改了网络接口后， 那就无法再以 ifdown eth0 的方式来关闭了！因为 ifdown 会分析比对目前的网络参数与 ifcfg-eth0 是否相符，不符的话，就会放弃该次动作。因此，使用 ifconfig 修改完毕后，应该要以 ifconfig eth0 down 才能够关闭该界面喔！

5.1.2 路由修改： route

我们在第二章网络基础的时候谈过关于路由的问题， 两部主机之间一定要有路由才能够互通 TCP/IP 的协议，否则就无法进行联机啊！一般来说，只要有网络接口， 该接口就会产生一个路由，所以我们安装的主机有一个 eth0 的接口，看起来就会是这样：

```
[root@www ~]# route [-nee]
[root@www ~]# route add [-net&#124;-host] [网域或主机] netmask [mask] [gw&#124;dev]
[root@www ~]# route del [-net&#124;-host] [网域或主机] netmask [mask] [gw&#124;dev]
观察的参数：
   -n  ：不要使用通讯协议或主机名，直接使用 IP 或 port number；
   -ee ：使用更详细的信息来显示
增加 (add) 与删除 (del) 路由的相关参数：
   -net    ：表示后面接的路由为一个网域；
   -host   ：表示后面接的为连接到单部主机的路由；
   netmask ：与网域有关，可以设定 netmask 决定网域的大小；
   gw      ：gateway 的简写，后续接的是 IP 的数值喔，与 dev 不同；
   dev     ：如果只是要指定由那一块网络卡联机出去，则使用这个设定，后面接 eth0 等
# 范例一：单纯的观察路由状态
[root@www ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
0.0.0.0         192.168.1.254   0.0.0.0         UG    0      0        0 eth0

[root@www ~]# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
link-local      *               255.255.0.0     U     1002   0        0 eth0
default         192.168.1.254   0.0.0.0         UG    0      0        0 eth0
```

由上面的例子当中仔细观察 route 与 route -n 的输出结果，你可以发现有加 -n 参数的主要是显示出 IP ，至于使用 route 而已的话，显示的则是『主机名』喔！也就是说，在预设的情况下， route 会去找出该 IP 的主机名，如果找不到呢？ 就会显示的钝钝的(有点小慢)，所以说，鸟哥通常都直接使用 route -n 啦！ 由上面看起来，我们也知道 default = 0.0.0.0/0.0.0.0 ， 而上面的信息有哪些你必须要知道的呢？

- Destination, Genmask：这两个玩意儿就是分别是 network 与 netmask 啦！所以这两个咚咚就组合成为一个完整的网域啰！
- Gateway：该网域是通过哪个 gateway 连接出去的？如果显示 0.0.0.0 表示该路由是直接由本机传送，亦即可以透过局域网络的 MAC 直接传讯；如果有显示 IP 的话，表示该路由需要经过路由器 (通讯闸) 的帮忙才能够传送出去。
- Flags：总共有多个旗标，代表的意义如下：
    - U (route is up)：该路由是启动的；
    - H (target is a host)：目标是一部主机 (IP) 而非网域；
    - G (use gateway)：需要透过外部的主机 (gateway) 来转递封包；
    - R (reinstate route for dynamic routing)：使用动态路由时，恢复路由信息的旗标；
    - D (dynamically installed by daemon or redirect)：已经由服务或转 port 功能设定为动态路由
    - M (modified from routing daemon or redirect)：路由已经被修改了；
    - ! (reject route)：这个路由将不会被接受(用来抵挡不安全的网域！)
- Iface：这个路由传递封包的接口。

此外，观察一下上面的路由排列顺序喔，依序是由小网域 (192.168.1.0/24 是 Class C)，逐渐到大网域 (169.254.0.0/16 Class B) 最后则是预设路由 (0.0.0.0/0.0.0.0)。 然后当我们要判断某个网络封包应该如何传送的时候，该封包会经由这个路由的过程来判断喔！ 举例来说，我上头仅有三个路由，若我有一个传往 192.168.1.20 的封包要传递，那首先会找 192.168.1.0/24 这个网域的路由，找到了！所以直接由 eth0 传送出去；

如果是传送到 Yahoo 的主机呢？ Yahoo 的主机 IP 是 119.160.246.241，我们通过判断 1)不是 192.168.1.0/24， 2)不是 169.254.0.0/16 结果到达 3)0/0 时，OK！传出去了，透过 eth0 将封包传给 192.168.1.254 那部 gateway 主机啊！所以说，路由是有顺序的。

因此当你重复设定多个同样的路由时， 例如在你的主机上的两张网络卡设定为相同网域的 IP 时，会出现什么情况？会出现如下的情况：

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
```

也就是说，由于路由是依照顺序来排列与传送的， 所以不论封包是由那个界面 (eth0, eth1) 所接收，都会由上述的 eth0 传送出去， 所以，在一部主机上面设定两个相同网域的 IP 本身没有什么意义！有点多此一举就是了。 除非是类似虚拟机 (Xen, VMware 等软件) 所架设的多主机时，才会有这个必要～

```
# 范例二：路由的增加与删除
[root@www ~]# route del -net 169.254.0.0 netmask 255.255.0.0 dev eth0
# 上面这个动作可以删除掉 169.254.0.0/16 这个网域！
# 请注意，在删除的时候，需要将路由表上面出现的信息都写入
# 包括 netmask , dev 等等参数喔！注意注意

[root@www ~]# route add -net 192.168.100.0 \
&gt; netmask 255.255.255.0 dev eth0
# 透过 route add 来增加一个路由！请注意，这个路由的设定必须要能够与你的网络互通。
# 举例来说，如果我下达底下的指令就会显示错误：
# route add -net 192.168.200.0 netmask 255.255.255.0 &lt;u&gt;gw 192.168.200.254&lt;/u&gt
# 因为我的主机内仅有 192.168.1.11 这个 IP ，所以不能直接与 192.168.200.254
# 这个网段直接使用 MAC 互通！这样说，可以理解吗？

[root@www ~]# route add default gw 192.168.1.250
# 增加预设路由的方法！请注意，只要有一个预设路由就够了喔！
# 同样的，那个 192.168.1.250 的 IP 也需要能与你的 LAN 沟通才行！
# 在这个地方如果你随便设定后，记得使用底下的指令重新设定你的网络
# &lt;u&gt;/etc/init.d/network restart&lt;/u&gt;
```

如果是要进行路由的删除与增加，那就得要参考上面的例子了，其实，使用 man route 里面的数据就很丰富了！仔细查阅一下啰！ 你只要记得，当出现『SIOCADDRT: Network is unreachable』 这个错误时，肯定是由于 gw 后面接的 IP 无法直接与你的网域沟通 (Gateway 并不在你的网域内)， 所以，赶紧检查一下是否输入错误啊！

Tips: 一般来说，鸟哥如果接触到一个新的环境内的主机，在不想要更动原系统的配置文件情况下，然后预计使用本书的网络环境设定时， 手动的处理就变成：『ifconfig eth0 192.168.1.100; route add default gw 192.168.1.254』这样就搞定了！ 直接联网与测试。等到完成测试后，再给她 /etc/init.d/network restart 恢复原系统的网络即可。



5.1.3 网络参数综合指令： ip

ip 是个指令喔！并不是那个 TCP/IP 的 IP 啦！这个 ip 指令的功能可多了！基本上，他就是整合了 ifconfig 与 route 这两个指令啰～不过， ip 可以达成的功能却又多更多！真是个相当厉害的指令。如果你有兴趣的话，请自行 vi /sbin/ifup ，就知道整个 ifup 就是利用 ip 这个指令来达成的。好了，如何使用呢？让我们来瞧一瞧先！

```
[root@www ~]# ip [option] [动作] [指令]
选项与参数：
option ：设定的参数，主要有：
    -s ：显示出该装置的统计数据(statistics)，例如总接受封包数等；
动作：亦即是可以针对哪些网络参数进行动作，包括有：
    link   ：关于装置 (device) 的相关设定，包括 MTU, MAC 地址等等
    addr/address ：关于额外的 IP 协议，例如多 IP 的达成等等；
    route ：与路由有关的相关设定
```

由上面的语法我们可以知道， ip 除了可以设定一些基本的网络参数之外，还能够进行额外的 IP 协议，包括多 IP 的达成，真是太完美了！底下我们就分三个部分 (link, addr, route) 来介绍这个 ip 指令吧！

- 关于装置接口 (device) 的相关设定： ip link

ip link 可以设定与装置 (device) 有关的相关参数，包括 MTU 以及该网络接口的 MAC 等等，当然也可以启动 (up) 或关闭 (down) 某个网络接口啦！整个语法是这样的：

```
[root@www ~]# ip [-s] link show  &lt;== 单纯的查阅该装置相关的信息
[root@www ~]# ip link set [device] [动作与参数]
选项与参数：
show：仅显示出这个装置的相关内容，如果加上 -s 会显示更多统计数据；
set ：可以开始设定项目， device 指的是 eth0, eth1 等等界面代号；
动作与参数：包括有底下的这些动作：
   up&#124;down   ：启动 (up) 或关闭 (down) 某个接口，其他参数使用默认的以太网络；
   address   ：如果这个装置可以更改 MAC 的话，用这个参数修改！
   name      ：给予这个装置一个特殊的名字；
   mtu       ：就是最大传输单元啊！

# 范例一：显示出所有的接口信息
[root@www ~]# ip link show
1: lo: &lt;LOOPBACK,UP,LOWER_UP&gt; mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: &lt;BROADCAST,MULTICAST,UP,LOWER_UP&gt; mtu 1500 qdisc pfifo_fast state UP qlen
    link/ether 08:00:27:71:85:bd brd ff:ff:ff:ff:ff:ff
3: eth1: &lt;BROADCAST,MULTICAST&gt; mtu 1500 qdisc noop state DOWN qlen 1000
    link/ether 08:00:27:2a:30:14 brd ff:ff:ff:ff:ff:ff
4: sit0: &lt;NOARP&gt; mtu 1480 qdisc noop state DOWN
    link/sit 0.0.0.0 brd 0.0.0.0

[root@www ~]# ip -s link show eth0
2: eth0: &lt;BROADCAST,MULTICAST,UP,LOWER_UP&gt; mtu 1500 qdisc pfifo_fast state UP qlen
    link/ether 08:00:27:71:85:bd brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    314685     3354     0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    27200      199      0       0       0       0
```

使用 ip link show 可以显示出整个装置接口的硬件相关信息，如上所示，包括网卡地址 (MAC)、MTU等等， 比较有趣的应该是那个 sit0 的接口了，那个 sit0 的界面是用在 IPv4 及 IPv6 的封包转换上的， 对于我们仅使用 IPv4 的网络是没有作用的。 lo 及 sit0 都是主机内部所自行设定的。 而如果加上 -s 的参数后，则这个网络卡的相关统计信息就会被列出来， 包括接收 (RX) 及传送 (TX) 的封包数量等等，详细的内容与 ifconfig 所输出的结果相同的。

```
# 范例二：启动、关闭与设定装置的相关信息
[root@www ~]# ip link set eth0 up
# 启动 eth0 这个装置接口；

[root@www ~]# ip link set eth0 down
# 阿就关闭啊！简单的要命～

[root@www ~]# ip link set eth0 mtu 1000
# 更改 MTU 的值，达到 1000 bytes，单位就是 bytes 啊！
```

更新网络卡的 MTU 使用 ifconfig 也可以达成啊！没啥了不起，不过，如果是要更改『网络卡代号、 MAC 地址的信息』的话，那可就得使用 ip 啰～不过，设定前可能得要先关闭该网络卡，否则会不成功。 如下所示：

```
# 范例三：修改网络卡代号、MAC 等参数
[root@www ~]# ip link set eth0 name vbird
SIOCSIFNAME: Device or resource busy
# 因为该装置目前是启动的，所以不能这样做设定。你应该要这样做：

[root@www ~]# ip link set eth0 down       &lt;==关闭界面
[root@www ~]# ip link set eth0 name vbird &lt;==重新设定
[root@www ~]# ip link show                &lt;==观察一下
2: vbird: &lt;BROADCAST,MULTICAST,UP,LOWER_UP&gt; mtu 1500 qdisc pfifo_fast state UP qlen
    link/ether 08:00:27:71:85:bd brd ff:ff:ff:ff:ff:ff
# 怕了吧！连网络卡代号都可以改变！不过，玩玩后记得改回来啊！
# 因为我们的 ifcfg-eth0 还是使用原本的装置代号！避免有问题，要改回来

[root@www ~]# ip link set vbird name eth0 &lt;==界面改回来

[root@www ~]# ip link set eth0 address aa:aa:aa:aa:aa:aa
[root@www ~]# ip link show eth0
# 如果你的网络卡支持硬件地址(MAC)可以更改的话，上面这个动作就可以更改
# 你的网络卡地址了！厉害吧！不过，还是那句老话，测试完之后请立刻改回来啊！
```

在这个装置的硬件相关信息设定上面，包括 MTU, MAC 以及传输的模式等等，都可以在这里设定。 有趣的是那个 address 的项目，那个项目后面接的可是硬件地址 (MAC) 而不是 IP 喔！ 很容易搞错啊！切记切记！更多的硬件参数可以使用 man ip 查阅一下与 ip link 有关的设定。

- 关于额外的 IP 相关设定： ip address

如果说 ip link 是与 OSI 七层协定 的第二层资料连阶层有关的话，那么 ip address (ip addr) 就是与第三层网络层有关的参数啦！ 主要是在设定与 IP 有关的各项参数，包括 netmask, broadcast 等等。

```
[root@www ~]# ip address show   &lt;==就是查阅 IP 参数啊！
[root@www ~]# ip address [add&#124;del] [IP参数] [dev 装置名] [相关参数]
选项与参数：
show     ：单纯的显示出接口的 IP 信息啊；
add&#124;del ：进行相关参数的增加 (add) 或删除 (del) 设定，主要有：
    IP 参数：主要就是网域的设定，例如 192.168.100.100/24 之类的设定喔；
    dev     ：这个 IP 参数所要设定的接口，例如 eth0, eth1 等等；
    相关参数：主要有底下这些：
        broadcast：设定广播地址，如果设定值是 + 表示『让系统自动计算』
        label    ：亦即是这个装置的别名，例如 eth0:0 就是了！
        scope    ：这个界面的领域，通常是这几个大类：
                   global ：允许来自所有来源的联机；
                   site   ：仅支持 IPv6 ，仅允许本主机的联机；
                   link   ：仅允许本装置自我联机；
                   host   ：仅允许本主机内部的联机；
                   所以当然是使用 global 啰！预设也是 global 啦！

# 范例一：显示出所有的接口之 IP 参数：
[root@www ~]# ip address show
1: lo: &lt;LOOPBACK,UP,LOWER_UP&gt; mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: &lt;BROADCAST,MULTICAST,UP,LOWER_UP&gt; mtu 1500 qdisc pfifo_fast state UP qlen
    link/ether 08:00:27:71:85:bd brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.100/24 brd 192.168.1.255 scope global eth0
    inet6 fe80::a00:27ff:fe71:85bd/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: &lt;BROADCAST,MULTICAST&gt; mtu 1500 qdisc noop state DOWN qlen 1000
    link/ether 08:00:27:2a:30:14 brd ff:ff:ff:ff:ff:ff
4: sit0: &lt;NOARP&gt; mtu 1480 qdisc noop state DOWN
    link/sit 0.0.0.0 brd 0.0.0.0
```

看到上面那个特殊的字体吗？没错！那就是 IP 参数啦！也是 ip address 最主要的功能。 底下我们进一步来新增虚拟的网络介面试看看：

```
# 范例二：新增一个接口，名称假设为 eth0:vbird
[root@www ~]# ip address add 192.168.50.50/24 broadcast + \
&gt; dev eth0 label eth0:vbird
[root@www ~]# ip address show eth0
2: eth0: &lt;BROADCAST,MULTICAST,UP,LOWER_UP&gt; mtu 1500 qdisc pfifo_fast state UP qlen
    link/ether 08:00:27:71:85:bd brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.100/24 brd 192.168.1.255 scope global eth0
    inet 192.168.50.50/24 brd 192.168.50.255 scope global eth0:vbird
    inet6 fe80::a00:27ff:fe71:85bd/64 scope link
       valid_lft forever preferred_lft forever
# 看到上面的特殊字体了吧？多出了一行新的接口，且名称是 eth0:vbird
# 至于那个 broadcast + 也可以写成 broadcast 192.168.50.255 啦！

[root@www ~]# ifconfig
eth0:vbird Link encap:Ethernet  HWaddr 08:00:27:71:85:BD
          inet addr:192.168.50.50  Bcast:192.168.50.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
# 如果使用 ifconfig 就能够看到这个怪东西了！可爱吧！ ^_^

# 范例三：将刚刚的界面删除
[root@www ~]# ip address del 192.168.50.50/24 dev eth0
# 删除就比较简单啊！ ^_^
```

- 关于路由的相关设定： ip route

这个项目当然就是路由的观察与设定啰！事实上， ip route 的功能几乎与 route 这个指令差不多，但是，他还可以进行额外的参数设计，例如 MTU 的规划等等，相当的强悍啊！

```
[root@www ~]# ip route show  &lt;==单纯的显示出路由的设定而已
[root@www ~]# ip route [add&#124;del] [IP或网域] [via gateway] [dev 装置]
选项与参数：
show ：单纯的显示出路由表，也可以使用 list ；
add&#124;del ：增加 (add) 或删除 (del) 路由的意思。
    IP或网域：可使用 192.168.50.0/24 之类的网域或者是单纯的 IP ；
    via      ：从那个 gateway 出去，不一定需要；
    dev      ：由那个装置连出去，这就需要了！
    mtu      ：可以额外的设定 MTU 的数值喔！

# 范例一：显示出目前的路由资料
[root@www ~]# ip route show
192.168.1.0/24 dev eth0  proto kernel  scope link  src 192.168.1.100
169.254.0.0/16 dev eth0  scope link  metric 1002
default via 192.168.1.254 dev eth0
```

如上表所示，最简单的功能就是显示出目前的路由信息，其实跟 route 这个指令相同啦！ 指示必须要注意几个小东西：

- proto：此路由的路由协议，主要有 redirect, kernel, boot, static, ra 等， 其中 kernel 指的是直接由核心判断自动设定。
- scope：路由的范围，主要是 link ，亦即是与本装置有关的直接联机。

再来看一下如何进行路由的增加与删除吧！

```
# 范例二：增加路由，主要是本机直接可沟通的网域
[root@www ~]# ip route add 192.168.5.0/24 dev eth0
# 针对本机直接沟通的网域设定好路由，不需要透过外部的路由器
[root@www ~]# ip route show
192.168.5.0/24 dev eth0  scope link
....(以下省略)....

# 范例三：增加可以通往外部的路由，需透过 router 喔！
[root@www ~]# ip route add 192.168.10.0/24 via 192.168.5.100 dev eth0
[root@www ~]# ip route show
192.168.5.0/24 dev eth0  scope link
....(其他省略)....
192.168.10.0/24 via 192.168.5.100 dev eth0
# 仔细看喔，因为我有 192.168.5.0/24 的路由存在 (我的网卡直接联系)，
# 所以才可以将 192.168.10.0/24 的路由丢给 192.168.5.100
# 那部主机来帮忙传递喔！与之前提到的 route 指令是一样的限制！

# 范例四：增加预设路由
[root@www ~]# ip route add default via 192.168.1.254 dev eth0
# 那个 192.168.1.254 就是我的预设路由器 (gateway) 的意思啊！ ^_^
# 眞的记得，只要一个预设路由就 OK ！

# 范例五：删除路由
[root@www ~]# ip route del 192.168.10.0/24
[root@www ~]# ip route del 192.168.5.0/24
```

事实上，这个 ip 的指令实在是太博大精深了！刚接触 Linux 网络的朋友，可能会看到有点晕～不要紧啦！你先会使用 ifconfig, ifup , ifdown 与 route 即可， 等以后有经验了之后，再继续回来玩 ip 这个好玩的指令吧！ ^_^ 有兴趣的话，也可以自行参考 ethtool 这个指令喔！(man ethtool)。

5.1.4 无线网络： iwlist, iwconfig

这两个指令你必须要有无线网卡才能够进行喔！这两个指令的用途是这样的：

- iwlist：利用无线网卡进行无线 AP 的侦测与取得相关的数据；
- iwconfig：设定无线网卡的相关参数。

这两个指令的应用我们在第四章里面的无线网卡设定谈了很多了， 所以这里我们不再详谈，有兴趣的朋友应该先使用 man iwlist 与 man iwconfig 了解一下语法， 然后再到前一章的无线网络小节查一查相关的用法，就了解了啦！ ^_^

5.1.5 手动使用 DHCP 自动取得 IP 参数： dhclient

如果你是使用 DHCP 协议在局域网络内取得 IP 的话，那么是否一定要去编辑 ifcfg-eth0 内的 BOOTPROTO 呢？ 嘿嘿！有个更快速的作法，那就是利用 dhclient 这个指令～因为这个指令才是真正发送 dhcp 要求工作的程序啊！那要如何使用呢？很简单！如果不考虑其他的参数，使用底下的方法即可：

```
[root@www ~]# dhclient eth0
```

够简单吧！这样就可以立刻叫我们的网络卡以 dhcp 协议去尝试取得 IP 喔！

5.2 网络侦错与观察指令

在网络的互助论坛中，最常听到的一句话就是：『高手求救！我的 Linux 不能连上网络了！』我的天呐！不能上网络的原因多的很！而要完全搞懂也不是一件简单的事情呢！ 不过，事实上我们可以自己使用测试软件来追踪可能的错误原因，而很多的网络侦测指令其实在 Linux 里头已经都预设存在了，只要你好好的学一学基本的侦测指令，那么一些朋友在告诉你如何侦错的时候， 你应该就立刻可以知道如何来搞定他啰！

其实我们在第四章谈到的五个检查步骤已经是相当详细的网络侦错流程了！ 只是还有些重要的侦测指令也得要来了解一下才好！

5.2.1 两部主机两点沟通： ping

这个 ping 是很重要的指令，ping 主要透过 ICMP 封包 来进行整个网络的状况报告，当然啦，最重要的就是那个 ICMP type 0, 8 这两个类型， 分别是要求回报与主动回报网络状态是否存在的特性。要特别注意的是， ping 还是需要透过 IP 封包来传送 ICMP 封包的， 而 IP 封包里面有个相当重要的 TTL 属性，这是很重要的一个路由特性， 详细的 IP 与 ICMP 表头资料请参考第二章网络基础的详细介绍。

```
[root@www ~]# ping [选项与参数] IP
选项与参数：
-c 数值：后面接的是执行 ping 的次数，例如 -c 5 ；
-n     ：在输出数据时不进行 IP 与主机名的反查，直接使用 IP 输出(速度较快)；
-s 数值：发送出去的 ICMP 封包大小，预设为 56bytes，不过你可以放大此一数值；
-t 数值：TTL 的数值，预设是 255，每经过一个节点就会少一；
-W 数值：等待响应对方主机的秒数。
-M [do&#124;dont] ：主要在侦测网络的 MTU 数值大小，两个常见的项目是：
   do   ：代表传送一个 DF (Don't Fragment) 旗标，让封包不能重新拆包与打包；
   dont：代表不要传送 DF 旗标，表示封包可以在其他主机上拆包与打包

# 范例一：侦测一下 168.95.1.1 这部 DNS 主机是否存在？
[root@www ~]# ping -c 3 168.95.1.1
PING 168.95.1.1 (168.95.1.1) 56(84) bytes of data.
64 bytes from 168.95.1.1: icmp_seq=1 ttl=245 time=15.4 ms
64 bytes from 168.95.1.1: icmp_seq=2 ttl=245 time=10.0 ms
64 bytes from 168.95.1.1: icmp_seq=3 ttl=245 time=10.2 ms

--- 168.95.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2047ms
rtt min/avg/max/mdev = 10.056/11.910/15.453/2.506 ms
```

ping 最简单的功能就是传送 ICMP 封包去要求对方主机回应是否存在于网络环境中，上面的响应消息当中，几个重要的项目是这样的：

- 64 bytes：表示这次传送的 ICMP 封包大小为 64 bytes 这么大，这是默认值， 在某些特殊场合中，例如要搜索整个网络内最大的 MTU 时，可以使用 -s 2000 之类的数值来取代；

- icmp_seq=1：ICMP 所侦测进行的次数，第一次编号为 1 ；
- ttl=243：TTL 与 IP 封包内的 TTL 是相同的，每经过一个带有 MAC 的节点 (node) 时，例如 router, bridge 时， TTL 就会减少一，预设的 TTL 为 255 ， 你可以透过 -t 150 之类的方法来重新设定预设 TTL 数值；
- time=15.4 ms：响应时间，单位有 ms(0.001秒)及 us(0.000001秒)， 一般来说，越小的响应时间，表示两部主机之间的网络联机越良好！

如果你忘记加上 -c 3 这样的规定侦测次数，那就得要使用 [ctrl]-c 将他结束掉了！

例题：写一支脚本程序 ping.sh ，透过这支脚本程序，你可以用 ping 侦测整个网域的主机是否有响应。此外，每部主机的侦测仅等待一秒钟，也仅侦测一次。答：由于仅侦测一次且等待一秒，因此 ping 的选项为： -W1 -c1 ，而位于本机所在的区网为 192.168.1.0/24 ，所以可以这样写 (vim /root/bin/ping.sh)：

```
#!/bin/bash
for siteip in $(seq 1 254)
do
    site="192.168.1.${siteip}"
    ping -c1 -W1 ${site} &&gt; /dev/null
    if [ "$?" == "0" ]; then
        echo "$site is UP"
    else
        echo "$site is DOWN"
    fi
done
```

特别注意一下，如果你的主机与待侦测主机并不在同一个网域内， 那么 TTL 预设使用 255 ，如果是同一个网域内，那么 TTL 预设则使用 64 喔！

- 用 ping 追踪路径中的最大 MTU 数值

  我们由第二章的网络基础里面谈到加大讯框 (frame) 时， 对于网络效能是有帮助的，因为封包打包的次数会减少，加上如果整个传输的媒体都能够接受这个 frame 而不需要重新进行封包的拆解与重组的话，那么效能当然会更好，那个修改 frame 大小的参数就是 MTU 啦！

  好了，现在我们知道网络卡的 MTU 修改可以透过 ifconfig 或者是 ip 等指令来达成，那么追踪整个网络传输的最大 MTU 时，又该如何查询？呵呵！最简单的方法当然是透过 ping 传送一个大封包， 并且不许中继的路由器或 switch 将该封包重组，那就能够处理啦！没错！可以这样的：

```
# 范例二：找出最大的 MTU 数值
[root@www ~]# ping -c 2 -s 1000 -M do 192.168.1.254
PING 192.168.1.254 (192.168.1.254) 1000(1028) bytes of data.
1008 bytes from 192.168.1.254: icmp_seq=1 ttl=64 time=0.311 ms
# 如果有响应，那就是可以接受这个封包，如果无响应，那就表示这个 MTU 太大了。

[root@www ~]# ping -c 2 -s 8000 -M do 192.168.1.254
PING 192.168.1.254 (192.168.1.254) 8000(8028) bytes of data.
From 192.168.1.100 icmp_seq=1 Frag needed and DF set (mtu = 1500)
# 这个错误讯息是说，本地端的 MTU 才到 1500 而已，你要侦测 8000 的 MTU
# 根本就是无法达成的！那要如何是好？用前一小节介绍的 ip link 来进行 MTU 设定吧！
```

不过，你需要知道的是，由于 IP 封包表头 (不含 options) 就已经占用了 20 bytes ，再加上 ICMP 的表头有 8 bytes ，所以当然你在使用 -s size 的时候，那个封包的大小就得要先扣除 (20+8=28) 的大小了。 因此如果要使用 MTU 为 1500 时，就得要下达『 ping -s 1472 -M do xx.yy.zz.ip 』才行啊！

另外，由于本地端的网络卡 MTU 也会影响到侦测，所以如果想要侦测整个传输媒体的 MTU 数值， 那么每个可以调整的主机就得要先使用 ifcofig 或 ip 先将 MTU 调大，然后再去进行侦测， 否则就会出现像上面提供的案例一样，可能会出现错误讯息的！

不过这个 MTU 不要随便调整啊！除非真的有问题。通常调整 MTU 的时间是在这个时候：

- 因为全部的主机群都是在内部的区网，例如丛集架构 (cluster) 的环境下， 由于内部的网络节点都是我们可以控制的，因此可以透过修改 MTU 来增进网络效能；
- 因为操作系统默认的 MTU 与你的网域不符，导致某些网站可以顺利联机，某些网站则无法联机。 以 Windows 操作系统作为联机分享的主机时，在 Client 端挺容易发生这个问题； 如果是要连上 Internet 的主机，注意不要随便调整 MTU ，因为我们无法知道 Internet 上面的每部机器能够支持的 MTU 到多大，因为......不是我们能够管的到的嘛 ^_^！ 另外，其实每种联机方式都有不同的 MTU 值，常见的各种接口的 MTU 值分别为：

```
| 网络接口 | MTU |
| --- | --- |
| Ethernet | 1500 |
| PPPoE | 1492 |
| Dial-up(Modem) | 576 |
```

5.2.2 两主机间各节点分析： traceroute

我们前面谈到的指令大多数都是针对主机的网络参数设定所需要的，而 ping 是两部主机之间的回声与否判断， 那么有没有指令可以追踪两部主机之间通过的各个节点 (node) 通讯状况的好坏呢？举例来说，如果我们联机到 yahoo 的速度比平常慢，你觉得是 (1)自己的网络环境有问题？ (2)还是外部的 Internet 有问题？如果是 (1) 的话，我们当然需要检查自己的网络环境啊，看看是否又有谁中毒了？但如果是 Internet 的问题呢？那只有『等等等』啊！ 判断是 (1) 还是 (2) 就得要使用 traceroute 这个指令啦！

```
[root@www ~]# traceroute [选项与参数] IP
选项与参数：
-n  ：可以不必进行主机的名称解析，单纯用 IP ，速度较快！
-U  ：使用 UDP 的 port 33434 来进行侦测，这是预设的侦测协议；
-I  ：使用 ICMP 的方式来进行侦测；
-T  ：使用 TCP 来进行侦测，一般使用 port 80 测试
-w  ：若对方主机在几秒钟内没有回声就宣告不治...预设是 5 秒
-p 埠号：若不想使用 UDP 与 TCP 的预设埠号来侦测，可在此改变埠号。
-i 装置：用在比较复杂的环境，如果你的网络接口很多很复杂时，才会用到这个参数；
         举例来说，你有两条 ADSL 可以连接到外部，那你的主机会有两个 ppp，
         你可以使用 -i 来选择是 ppp0 还是 ppp1 啦！
-g 路由：与 -i 的参数相仿，只是 -g 后面接的是 gateway 的 IP 就是了。

# 范例一：侦测本机到 yahoo 去的各节点联机状态
[root@www ~]# traceroute -n tw.yahoo.com
traceroute to tw.yahoo.com (119.160.246.241), 30 hops max, 40 byte packets
 1  192.168.1.254  0.279 ms  0.156 ms  0.169 ms
 2  172.20.168.254  0.430 ms  0.513 ms  0.409 ms
 3  10.40.1.1  0.996 ms  0.890 ms  1.042 ms
 4  203.72.191.85  0.942 ms  0.969 ms  0.951 ms
 5  211.20.206.58  1.360 ms  1.379 ms  1.355 ms
 6  203.75.72.90  1.123 ms  0.988 ms  1.086 ms
 7  220.128.24.22  11.238 ms  11.179 ms  11.128 ms
 8  220.128.1.82  12.456 ms  12.327 ms  12.221 ms
 9  220.128.3.149  8.062 ms  8.058 ms  7.990 ms
10  * * *
11  119.160.240.1  10.688 ms  10.590 ms 119.160.240.3  10.047 ms
12  * * * &lt;==可能有防火墙装置等情况发生所致
```

这个 traceroute 挺有意思的，这个指令会针对欲连接的目的地之所有 node 进行 UDP 的逾时等待， 例如上面的例子当中，由鸟哥的主机连接到 Yahoo 时，他会经过 12 个节点以上，traceroute 会主动的对这 12 个节点做 UDP 的回声等待，并侦测回复的时间，每节点侦测三次，最终回传像上头显示的结果。 你可以发现每个节点其实回复的时间大约在 50 ms 以内，算是还可以的 Internet 环境了。

比较特殊的算是第 10/12 个，会回传星号的，代表该 node 可能设有某些防护措施，让我们发送的封包信息被丢弃所致。 因为我们是直接透过路由器转递封包，并没有进入路由器去取得路由器的使用资源，所以某些路由器仅支持封包转递， 并不会接受来自客户端的各项侦测啦！此时就会出现上述的问题。因为 traceroute 预设使用 UDP 封包，如果你想尝试使用其他封包， 那么 -I 或 -T 可以试看看啰！

由于目前 UDP/ICMP 的攻击层出不穷，因此很多路由器可能就此取消这两个封包的响应功能。所以我们可以使用 TCP 来侦测呦！ 例如使用同样的方法，透过等待时间 1 秒，以及 TCP 80 埠口的情况下，可以这样做：

```
[root@www ~]# traceroute -w 1 -n -T tw.yahoo.com
```

5.2.3 察看本机的网络联机与后门： netstat

如果你觉得你的某个网络服务明明就启动了，但是就是无法造成联机的话，那么应该怎么办？ 首先你应该要查询一下自己的网络接口所监听的端口 (port) 来看看是否真的有启动，因为有时候屏幕上面显示的 [OK] 并不一定是 OK 啊！ ^_^

```
[root@www ~]# netstat -[rn]       &lt;==与路由有关的参数
[root@www ~]# netstat -[antulpc]  &lt;==与网络接口有关的参数
选项与参数：
与路由 (route) 有关的参数说明：
-r  ：列出路由表(route table)，功能如同 route 这个指令；
-n  ：不使用主机名与服务名称，使用 IP 与 port number ，如同 route -n
与网络接口有关的参数：
-a  ：列出所有的联机状态，包括 tcp/udp/unix socket 等；
-t  ：仅列出 TCP 封包的联机；
-u  ：仅列出 UDP 封包的联机；
-l  ：仅列出有在 Listen (监听) 的服务之网络状态；
-p  ：列出 PID 与 Program 的檔名；
-c  ：可以设定几秒钟后自动更新一次，例如 -c 5 每五秒更新一次网络状态的显示；

# 范例一：列出目前的路由表状态，且以 IP 及 port number 显示：
[root@www ~]# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.1.0     0.0.0.0         255.255.255.0   U         0 0          0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U         0 0          0 eth0
0.0.0.0         192.168.1.254   0.0.0.0         UG        0 0          0 eth0
# 其实这个参数就跟 route -n 一模一样，对吧！这不是 netstat 的主要功能啦！

# 范例二：列出目前的所有网络联机状态，使用 IP 与 port number
[root@www ~]# netstat -an
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address     Foreign Address     State
....(中间省略)....
tcp        0      0 127.0.0.1:25      0.0.0.0:*           LISTEN
tcp        0     52 192.168.1.100:22  192.168.1.101:1937  ESTABLISHED
tcp        0      0 :::22             :::*                LISTEN
....(中间省略)....
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags    Type    State      I-Node Path
unix  2      [ ACC ]  STREAM  LISTENING  11075  @/var/run/hald/dbus-uukdg1qMPh
unix  2      [ ACC ]  STREAM  LISTENING  10952  /var/run/dbus/system_bus_socket
unix  2      [ ACC ]  STREAM  LISTENING  11032  /var/run/acpid.socket
....(底下省略)....
```

netstat 的输出主要分为两大部分，分别是 TCP/IP 的网络接口部分，以及传统的 Unix socket 部分。 还记得我们在基础篇里面曾经谈到档案的类型吗？那个 socket 与 FIFO 档案还记得吧？ 那就是在 Unix 接口用来做为程序数据交流的接口了，也就是上头表格内看到的 Active Unix domain sockets 的内容啰～

通常鸟哥都是建议加上『 -n 』这个参数的，因为可以避过主机名与服务名称的反查，直接以 IP 及端口号码 (port number) 来显示，显示的速度上会快很多！至于在输出的讯息当中， 我们先来谈一谈关于网络联机状态的输出部分，他主要是分为底下几个大项：

- Proto：该联机的封包协议，主要为 TCP/UDP 等封包；
- Recv-Q：非由用户程序连接所复制而来的总 bytes 数；
- Send-Q：由远程主机所传送而来，但不具有 ACK 标志的总 bytes 数， 意指主动联机 SYN 或其他标志的封包所占的 bytes 数；
- Local Address：本地端的地址，可以是 IP (-n 参数存在时)， 也可以是完整的主机名。使

用的格是就是『 IP:port 』只是 IP 的格式有 IPv4 及 IPv6 的差异。 如上所示，在 port 22 的接口中，使用的 :::22 就是针对 IPv6 的显示，事实上他就相同于 0.0.0.0:22 的意思。 至于 port 25 仅针对 lo 接口开放，意指 Internet 基本上是无法连接到我本机的 25 埠口啦！

- Foreign Address：远程的主机 IP 与 port number
- stat：状态栏，主要的状态含有：
    - ESTABLISED：已建立联机的状态；
    - SYN_SENT：发出主动联机 (SYN 标志) 的联机封包；
    - SYN_RECV：接收到一个要求联机的主动联机封包；
    - FIN_WAIT1：该插槽服务(socket)已中断，该联机正在断线当中；
    - FIN_WAIT2：该联机已挂断，但正在等待对方主机响应断线确认的封包；
    - TIME_WAIT：该联机已挂断，但 socket 还在网络上等待结束；
    - LISTEN：通常用在服务的监听 port ！可使用『 -l 』参数查阅。

基本上，我们常常谈到的 netstat 的功能，就是在观察网络的联机状态了，而网络联机状态中， 又以观察『我目前开了多少的 port 在等待客户端的联机』以及 『目前我的网络联机状态中，有多少联机已建立或产生问题』最常见。 那你如何了解与观察呢？通常鸟哥是这样处理的：

```
# 范例三：秀出目前已经启动的网络服务
[root@www ~]# netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address  Foreign Address   State   PID/Program name
tcp        0      0 0.0.0.0:34796  0.0.0.0:*         LISTEN 987/rpc.statd
tcp        0      0 0.0.0.0:111    0.0.0.0:*         LISTEN 969/rpcbind
tcp        0      0 127.0.0.1:25   0.0.0.0:*         LISTEN 1231/master
tcp        0      0 :::22          :::*              LISTEN 1155/sshd
udp        0      0 0.0.0.0:111    0.0.0.0:*                969/rpcbind
....(底下省略)....
# 上面最重要的其实是那个 -l 的参数，因为可以仅列出有在 Listen 的 port
```

你可以发现很多的网络服务其实仅针对本机的 lo 开放而已，因特网是连接不到该埠口与服务的。 而由上述的数据我们也可以看到，启动 port 111 的，其实就是 rpcbind 那只程序，那如果想要关闭这个埠口， 你可以使用 kill 删除 PID 969，也可以使用 killall 删除 rpcbind 这个程序即可。如此一来， 很轻松的你就能知道哪个程序启动了哪些端口啰！

```
# 范例四：观察本机上头所有的网络联机状态
[root@www ~]# netstat -atunp
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address      Foreign Address     State       PID/Program
tcp        0      0 0.0.0.0:111        0.0.0.0:*           LISTEN      969/rpcbind
tcp        0      0 0.0.0.0:22         0.0.0.0:*           LISTEN      1155/sshd
tcp        0      0 127.0.0.1:25       0.0.0.0:*           LISTEN      1231/master
tcp        0     52 192.168.1.100:22   192.168.1.101:1937  ESTABLISHED 4716/0
....(底下省略)....
```

看到上头的特殊字体吧？那代表目前已经建立联机的一条网络联机，他是由远程主机 192.168.1.101 启动一个大于 1024 的埠口向本地端主机 192.168.1.100 的 port 22 进行的一条联机， 你必须要想起来的是：『Client 端是随机取一个大于 1024 以上的 port 进行联机』，此外『只有 root 可以启动小于 1024 以下的 port 』，那就看的懂上头那条联机啰！如果这条联机你想要砍掉他的话， 看到最右边的 4716 了没？ kill 会用吧！ ^_^

至于传统的 Unix socket 的数据，记得使用 man netstat 查阅一下吧！ 这个 Unix socket 通常是用在一些仅在本机上运作的程序所开启的插槽接口文件， 例如 X Window 不都是在本机上运作而已吗？那何必启动网络的 port 呢？当然可以使用 Unix socket 啰，另外，例如 Postfix 这一类的网络服务器，由于很多动作都是在本机上头来完成的， 所以以会占用很多的 Unix socket 喔！

例题：请说明服务名称与 port number 的对应在 Linux 当中，是用那个档案来设定对应的？
答：/etc/services

5.2.4 侦测主机名与 IP 对应： host, nslookup

关于主机名与 IP 的对应中，我们主要介绍的是 DNS 客户端功能的 dig 这个指令。不过除了这个指令之外， 其实还有两个更简单的指令，那就是 host 与 nslookup 啦！底下让我们来聊聊这两个指令吧！

- host

这个指令可以用来查出某个主机名的 IP 喔！举例来说，我们想要知道 tw.yahoo.com 的 IP 时，可以这样做：

```
[root@www ~]# host [-a] hostname [server]
选项与参数：
-a ：列出该主机详细的各项主机名设定数据
[server] ：可以使用非为 /etc/resolv.conf 的 DNS 服务器 IP 来查询。

# 范例一：列出 tw.yahoo.com 的 IP
[root@www ~]# host tw.yahoo.com
tw.yahoo.com is an alias for tw-cidr.fyap.b.yahoo.com.
tw-cidr.fyap.b.yahoo.com is an alias for tw-tpe-fo.fyap.b.yahoo.com.
tw-tpe-fo.fyap.b.yahoo.com has address 119.160.246.241
```

瞧！IP 是 119.160.246.241 啊！很简单就可以查询到 IP 了！那么这个 IP 是向谁查询的呢？其实就是写在 /etc/resolv.conf 那个档案内的 DNS 服务器 IP 啦！如果不想要使用该档案内的主机来查询，也可以这样做：

```
[root@www ~]# host tw.yahoo.com 168.95.1.1
Using domain server:
Name: 168.95.1.1
Address: 168.95.1.1#53
Aliases:

tw.yahoo.com is an alias for tw-cidr.fyap.b.yahoo.com.
tw-cidr.fyap.b.yahoo.com is an alias for tw-tpe-fo.fyap.b.yahoo.com.
tw-tpe-fo.fyap.b.yahoo.com has address 119.160.246.241
```

会告诉我们所使用来查询的主机是哪一部呐！这样就够清楚了吧！不过，再怎么清楚也比不过 dig 这个指令的，所以这个指令仅是参考参考啦！

- nslookup

这玩意儿的用途与 host 基本上是一样的，就是用来作为 IP 与主机名对应的检查， 同样是使用 /etc/resolv.conf 这个档案来作为 DNS 服务器的来源选择。

```
[root@www ~]# nslookup [-query=[type]] [hostname&#124;IP]
选项与参数：
-query=type：查询的类型，除了传统的 IP 与主机名对应外，DNS 还有很多信息，
             所以我们可以查询很多不同的信息，包括 mx, cname 等等，
             例如： -query=mx 的查询方法！

# 范例一：找出 www.google.com 的 IP
[root@www ~]# nslookup www.google.com
Server:         168.95.1.1
Address:        168.95.1.1#53

Non-authoritative answer:
www.google.com  canonical name = www.l.google.com.
Name:   www.l.google.com
Address: 74.125.71.106
....(底下省略)....

# 范例二：找出 168.95.1.1 的主机名
[root@www ~]# nslookup 168.95.1.1
Server:         168.95.1.1
Address:        168.95.1.1#53

1.1.95.168.in-addr.arpa name = dns.hinet.net.
```

如何，看起来与 host 差不多吧！不过，这个 nslookup 还可以由 IP 找出主机名喔！ 例如那个范例二，他的主机名是： dns.hinet.net 哩！目前大家都建议使用 dig 这个指令来取代 nslookup ，我们会在第十九章 DNS 服务器那时再来好好谈一谈吧！

5.3 远程联机指令与实时通讯软件

啥是远程联机呢？其实就是在不同的计算机之间进行登入的情况啦！我们可以透过 telnet, ssh 或者是 ftp 等协议来进行远程主机的登入。底下我们就分别来介绍一下这些基本的指令吧！这里仅是谈到客户端功能喔， 相关的服务器我们则会在后续进行说明的。

5.3.1 终端机与 BBS 联机： telnet

telnet 是早期我们在个人计算机上面要链接到服务器工作时，最重要的一个软件了！他不但可以直接连接到服务器上头， 还可以用来连结 BBS 呢！非常棒！不过， telnet 本身的数据在传送的时候是使用明码 (原始的数据，没有加密) ， 所以数据在 Internet 上面跑的时候，会比较危险一点 (就怕被别人监听啊)。 更详细的资料我们会在第十一章远程联机服务器内做介绍的。

```
[root@www ~]# telnet [host&#124;IP [port]]

# 范例一：连结到台湾相当热门的 PTT BBS 站 ptt.cc
[root@www ~]# yum install telnet  &lt;==默认没有安装这软件
[root@www ~]# telnet ptt.cc
    欢迎来到 批踢踢实业坊 目前有【100118】名使用者与您一同对抗炎炎夏日。

请输入代号，或以 guest 参观，或以 new 注册:
[高手召集令] 台湾黑客年会 暑假与你骇翻南港 http://reg.hitcon.org/hit2011
要学计算机，首选台湾大学信息训练班!  http://tinyurl.com/3z42apw
```

如上所示，我们可以透过 telnet 轻易的连结到 BBS 上面，而如果你的主机有开启 telnet 服务器服务的话，同样的利用『 telnet IP 』并且输入账号与密码之后，就能够登入主机了。 另外，在 Linux 上的 telnet 软件还提供了 Kerberos 的认证方式，有兴趣的话请自行参阅 man telnet 的说明。

除了连结到服务器以及连结到 BBS 站之外， telnet 还可以用来连结到某个 port (服务) 上头呐！ 举例来说，我们可以用 telnet 连接到 port 110 ，看看这个 port 是否有正确的启动呢？

```
# 范例二：侦测本机端的 110 这个 port 是否正确启动？
[root@www ~]# telnet localhost 110
Trying 127.0.0.1...
telnet: connect to address 127.0.0.1: Connection refused
# 如果出现这样的讯息，代表这个 port 没有启动或者是这个联机有问题，
# 因为你看到那个 refused 嘛！

[root@www ~]# telnet localhost 25
Trying ::1...
Connected to localhost.
Escape character is '^]'.
220 www.centos.vbird ESMTP Postfix
ehlo localhost
250-www.centos.vbird
250-PIPELINING
250-SIZE 10240000
....(中间省略)....
250 DSN
quit
221 2.0.0 Bye
Connection closed by foreign host.
```

瞧！根据输出的结果，我们就能够知道这个通讯协议 (port number 提供的通讯协议功能) 是否有成功的启动呐！ 而在每个 port 所监听的服务都有其特殊的指令，例如上述的 port 25 就是在本机接口所提供的电子邮件服务， 那个服务所支持的指令就如同上面使用的数据一样，但是其他的 port 就不见得支持这个『 ehlo 』的命令， 因为不同的 port 有不同的程序嘛！所以当然支持的命令就不同啰！

5.3.2 FTP 联机软件： ftp, lftp

现在的人们由于有高容量的 email 可以用，因此传送档案可以很轻松的透过 email 。不过 email 还是有单封信件容量限制， 如果想要一口气传送个几百 MB 的档案，恐怕还是得要透过 FTP 这个通讯协议才行啊！文字接口的 FTP 软件主要有 ftp, lftp 两个，图形接口的呢？在 CentOS 上面预设有 gftp 这个好用的东东。在这里我们仅介绍文字接口的两个指令而已。

- ftp

ftp 这个指令很简单，用在处理 FTP 服务器的下载数据啦。由于鸟哥所在的位置在昆山科大，因此这里使用昆山科大的 FTP 服务器为例：

```
[root@www ~]# ftp [host&#124;IP] [port]

# 范例一：联机到昆山科大去看看
[root@www ~]# yum install ftp
[root@www ~]# ftp ftp.ksu.edu.tw
Connected to ftp.ksu.edu.tw (120.114.150.21).
220---------- Welcome to Pure-FTPd [privsep] ----------
220-You are user number 1 of 50 allowed.
220-Local time is now 16:25\. Server port: 21.
220-Only anonymous FTP is allowed here  &lt;==讯息要看啊！这个 FTP 仅支援匿名
220-IPv6 connections are also welcome on this server.
220 You will be disconnected after 5 minutes of inactivity.
Name (ftp.ksu.edu.tw:root): anonymous  &lt;==鸟哥这里用匿名登录！
230 Anonymous user logged in            &lt;==嗯！确实是匿名登录了！
Remote system type is UNIX.
Using binary mode to transfer files.
ftp&gt;                &lt;==最终登入的结果看起来是这样！
ftp&gt; help           &lt;==提供需要的指令说明，可以常参考！
ftp&gt; dir            &lt;==显示远程服务器的目录内容 (文件名列表)
ftp&gt; cd /pub        &lt;==变换目录到 /pub 当中
ftp&gt; get filename   &lt;==下载单一档案，档名为 filename
ftp&gt; mget filename* &lt;==下载多个档案，可使用通配符 *
ftp&gt; put filename   &lt;==上传 filename 这个档案到服务器上
ftp&gt; delete file    &lt;==删除主机上的 file 这个档案
ftp&gt; mkdir dir      &lt;==建立 dir 这个目录
ftp&gt; lcd /home      &lt;==切换『本地端主机』的工作目录
ftp&gt; passive        &lt;==启动或关闭 passive 模式
ftp&gt; binary         &lt;==数据传输模式设定为 binary 格式
ftp&gt; bye            &lt;==结束 ftp 软件的使用
```

FTP 其实算是一个很麻烦的协议，因为他使用两个 port 分别进行命令与数据的交流，详细的数据我们会在第二十一章的 FTP 服务器内详谈，这里我们先单纯的介绍一下如何使用 ftp 这个软件。首先我们当然是需要登入啰， 所以在上头的表格当中我们当然需要填入账号与密码了。不过由于昆山科大仅提供匿名登录，而匿名登录者的账号就是『 anonymous 』所以直接填写那个账号即可。如果是私人的 FTP 时，才需要提供一组完整的账号与密码啦！

登入 FTP 主机后，就能够使用 ftp 软件的功能进行上传与下载的动作，几个常用的 ftp 内指令如上表，不过，鸟哥建议你可以连到大学的 FTP 网站后，使用 help (或问号 ?) 来参考可用的指令，然后尝试下载以测试使用一下这个指令吧！这样以后没有浏览器的时候，你也可以到 ftp 下载了呢！不错吧！另外你要注意的是，离开 ftp 软件时，得要输入『 bye 』喔！不是『 exit 』啦！

如果由于某些理由，让你的 FTP 主机的 port 开在非正规的埠口时，那你就可以利用底下的方式来连接到该部主机喔！

```
[root@www ~]# ftp hostname 318
# 假设对方主机的 ftp 服务开启在 318 这个 port 啊！
```

- lftp (自动化脚本)

单纯使用 ftp 总是觉得很麻烦，有没有更快速的 ftp 用户软件呢？让我们可以使用类似网址列的方式来登入 FTP 服务器啊？有的，那就是 lftp 的功能了！ lftp 预设使用匿名登录 FTP 服务器，可以使用类似网址列的方式取得数据， 使用上比单纯的 ftp 要好用些。此外，由于可在指令列输入账号/密码，可以辅助进行程序脚本的设计喔！

```
[root@www ~]# lftp [-p port] [-u user[,pass]] [host&#124;IP]
[root@www ~]# lftp -f filename
[root@www ~]# lftp -c "commands"
选项与参数：
-p  ：后面可以直接接上远程 FTP 主机提供的 port
-u  ：后面则是接上账号与密码，就能够连接上远程主机了
      如果没有加账号密码， lftp 默认会使用 anonymous 尝试匿名登录
-f  ：可以将指令写入脚本中，这样可以帮助进行 shell script 的自动处理喔！
-c  ：后面直接加上所需要的指令。

# 范例一：利用 lftp 登入昆山科大的 FTP 服务器
[root@www ~]# yum install lftp
[root@www ~]# lftp ftp.ksu.edu.tw
lftp ftp.ksu.edu.tw:~&gt;
# 瞧！一下子就登入了！很快乐吧！ ^_^！你同样可使用 help 去查阅相关内部指令
```

至于登入 FTP 主机后，一样可以使用『help』来显示出可以执行的指令，与 ftp 很类似啦！不过多了书签的功能，而且也非常的类似 bash 呐！很不错哟！除了这个好用的文字接口的 FTP 软件之外，事实上还有很多图形接口的好用软件呢！ 最常见的就是 gftp 了，非常的容易上手喔！ CentOS 本身就有提供 gftp 了，你可以拿出原版的光盘来安装，然后进入 X Window 后， 启动一个 shell ，输入『 gftp 』就能够发现他的好用啦！

如果你想要定时的去捉下昆山科大 FTP 网站下的 /pub/CentOS/RPM-GPG* 的档案时，那么那个脚本应该要怎么写呢？ 我们尝试来写写看吧！

```
# 使用档案配合 lftp 去处理时：
[root@www ~]# mkdir lftp; cd lftp
[root@www lftp]# vim lftp.ksu.sh
open ftp.ksu.edu.tw
cd /pub/CentOS/
mget -c -d RPM-GPG*
bye
[root@www lftp]# lftp -f lftp.ksu.sh
[root@www lftp]# ls
lftp.ksu.sh      RPM-GPG-KEY-CentOS-3 RPM-GPG-KEY-CentOS-4 RPM-GPG-KEY-CentOS-6
RPM-GPG-KEY-beta RPM-GPG-KEY-centos4  RPM-GPG-KEY-CentOS-5

# 直接将要处理的动作加入 lftp 指令中
[root@www lftp]# vim lftp.ksu.sh
lftp -c "open ftp.ksu.edu.tw
cd /pub/CentOS/
mget -c -d RPM-GPG*
bye"
[root@www lftp]# sh lftp.ksu.sh
```

若为非匿名登录时，则可以使用『 open -u username,password hostname 』修改 lftp.ksu.sh 的第一行！ 如果再将这个脚本写入 crontab 当中，你就可以定时的以 FTP 进行上传/下载的功能啰！这就是文字指令的好处！

5.3.3 图形接口的实时通讯软件： pidgin (gaim 的延伸)

现在应该大家都知道什么是 MSN, 雅虎实时通以及其他的通讯软件吧？那么要连上这些服务器时，该怎么处理哪？很简单，在 X Window 底下使用 pidgin 就好了！简直简单到不行～请先进入 X Window 系统，然后经过『应用程序』--> 『因特网』-->『Pidgin 网络实时通』启动他即可 (请注意你必须已经安装了 pidgin 了，可用 yum install pidgin 处理)。

不过，伤脑筋的是，我们所安装的 basic server 类型的 CentOS 6.x 主要做为服务器之用，所以连图形接口也没有给我们。 所以，鸟哥又用另外一部主机安装成 Desktop 的模式，利用该部主机来测试 pidgin 这玩意儿的！因此， 底下的练习你也可以先略过，等到你安装另一部 Desktop linux 时再来玩玩！



图 5.3-1、pidgin 的欢迎画面

在上图中按下『新增』，然后你会看到如下的画面：



图 5.3-2、pidgin 支持的实时通讯数据

很神奇的是，pidgin 支持的通讯有够多的！我们使用 MSN 来作个解释好了：

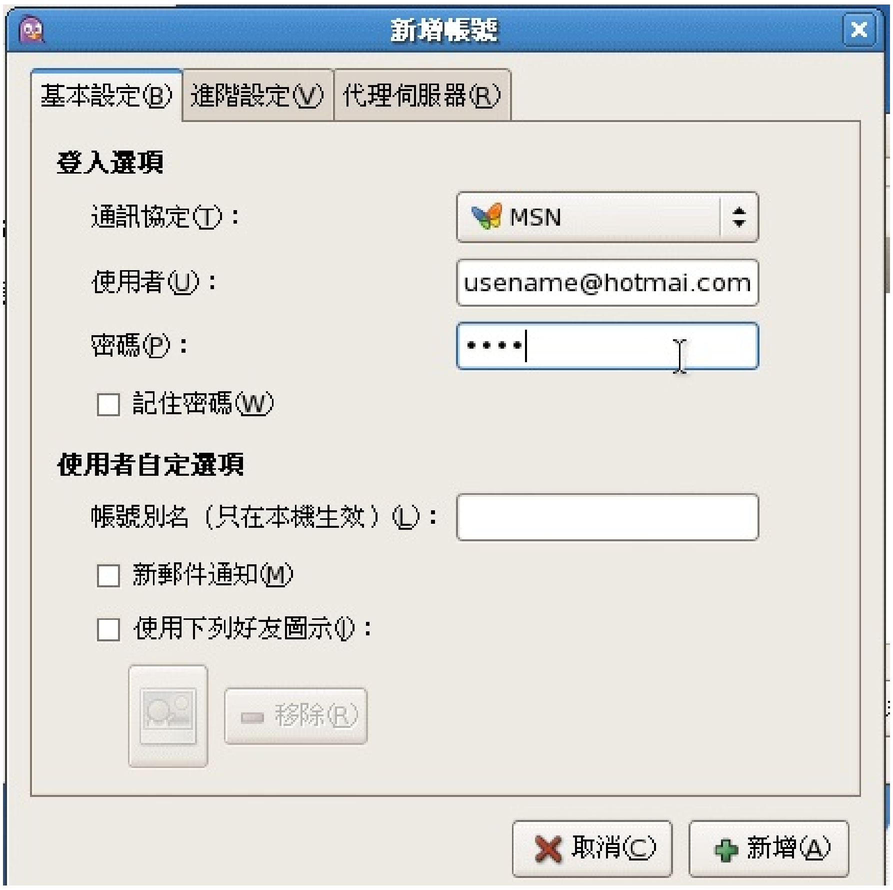


图 5.3-3、设定 MSN 的账号示意图

如上图，在画面中输入你的账号与密码，如果是在公用的计算机上，千万不要按下『记住密码』项目喔！按下新增后， pidgin 预设就会尝试登入了！登入后的画面如下所示：



图 5.3-4、使用 pidgin 的 MSN 方式进行连天啰

如果想要注销了，那么就按下图 5.3-4 最右边那个窗口，将『启动』的那个方框勾选取消，你就直接注销啰！

5.4 文字接口网页浏览

什么？文字界面竟然有浏览器！别逗了好不好？呵呵！谁有那个时间在逗你呦！真的啦！有这个东西， 是在文字界面下上网浏览的好工具！分别是 links 及 wget 这两个宝贝蛋，但是，你必需要确定你已经安装了这两个套件才行。 好佳在的是，CentOS 预设这两个玩意儿都有安装喔！底下就让我们来聊一聊这两个好用的家伙吧！

5.4.1 文字浏览器：links

其实早期鸟哥最常使用的是 lynx 这个文字浏览器，不过 CentOS 从 5.x 以后默认使用的文字浏览器是 links 这一支，这两支的使用方式又非常的类似，因此，在这一版当中，我们就仅介绍 links 啰！若对 lynx 有兴趣的话， 自己 man 一下吧！

这个指令可以让我们来浏览网页，但鸟哥认为，这个档案最大的功能是在『 查阅 Linux 本机上面以 HTML 语法写成的文件数据 (document)』 怎么说呢？如果你曾经到 Linux 本机底下的 /usr/share/doc 这个目录看过文件数据的话， 就会常常发现一些网页档案，使用 vi 去查阅时，老是看到一堆 HTML 的语法！有碍阅读啊～ 这时候使用 links 就是个好方法啦！可以看的清清楚楚啊！ ^_^

```
[root@www ~]# links [options] [URL]
选项与参数：
-anonymous [0&#124;1]：是否使用匿名登录的意思；
-dump [0&#124;1]      ：是否将网页的数据直接输出到 standard out 而非 links 软件功能
-dump_charset    ：后面接想要透过 dump 输出到屏幕的语系编码，big5 使用 cp950 喔

# 范例一：浏览 Linux kernel 网站
[root@www ~]# links http://www.kernel.org
```

当我直接输入 links 网站网址后，就会出现如下的图示：

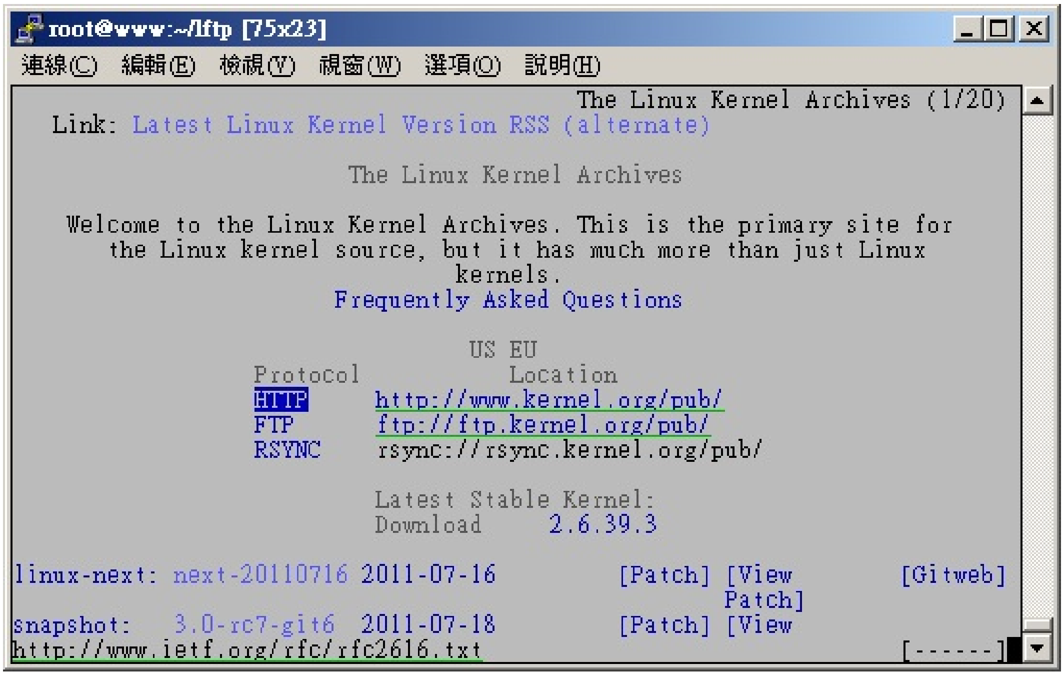


图 5.4-1、使用 links 查询网页数据的显示结果

上面这个画面的基本说明如下：

- 进入画面之后，由于是文字型态，所以编排可能会有点位移！不过不打紧！不会影响我们看咚咚！
- 这个时候可以使用『上下键』来让光标在上面的选项当中(如信箱、书签等等的)，按下 Enter 就进入该页面
- 可以使用『左右键』来移动『上一页或下一页』
- 一些常见功能按键：
  - h：history ，曾经浏览过的 URL 就显示到画面中
  - g：Goto URL，按 g 后输入网页地址(URL) 如 :http://www.abc.edu/等
  - d：download，将该链接数据下载到本机成为档案；
  - q：Quit，离开 links 这个软件；
  - o：Option，进入功能参数的设定值修改中，最终可写入 ~/.elinks/elinks.conf 中
  - Ctrl+C ：强迫切断 links 的执行。
  - 箭头键:
    - 上 ：移动光标至本页中 "上一个可连结点" .
    - 下 ：移动光标至本页中 "下一个可连结点" .
    - 左 ：back. 跳回上一页.
    - 右 ：进入反白光标所链接之网页.
    - ENTER 同鼠标 "右" 键.

至于如果是浏览 Linux 本机上面的网页档案，那就可以使用如下的方式：

```
[root@www ~]# links /usr/share/doc/HTML/index.html
```

在鸟哥的 CentOS 6.x 当中，有这么一个档案，我就可以利用 links 来取出察看呐！显示的结果有点像底下这样：

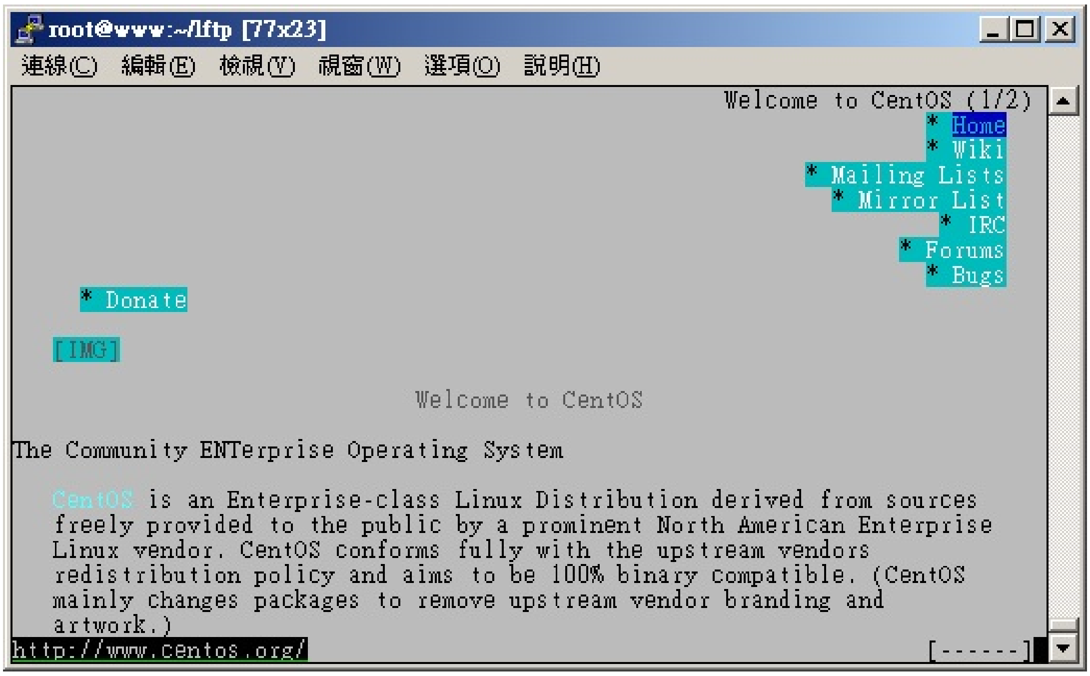


图 5.4-2、使用 links 查询本机的 HTML 文件档案

当然啦！因为你的环境可能是在 Linux 本机的 tty1~tty6 ，所以无法显示出中文，这个时候你就得要设定为：『LANG=en_US』之类的语系设定才行喔！另外，如果某些时刻你必须上网点选某个网站以自动取得更新时。举例来说，早期的自动在线更新主机名系统，仅支持网页更新，那你如何进行更新呢？嘿嘿！可以使用 links 喔！利用 -dump 这个参数处理先：

```
# 透过 links 将 tw.yahoo.com 的网页内容整个抓下来储存
[root@www ~]# links -dump http://tw.yahoo.com &gt; yahoo.html

# 某个网站透过 GET 功能可以上传账号为 user 密码为 pw ，用文字接口处理为：
[root@www ~]# links -dump \
&gt; http://some.site.name/web.php?name=user&password=pw &gt; testfile
```

上面的网站后面有加个问号 (?) 对吧？后面接的则是利用网页的『 GET 』功能取得的各项变量数据， 利用这个功能，我们就可以直接点选到该网站上啰！非常的方便吧！而且会将执行的结果输出到 testfile 档案中，不过如果网站提供的数据是以『 POST 』为主的话，那鸟哥就不知道如何搞定了。 GET 与 POST 是 WWW 通讯协议中，用来将数据透过浏览器上传到服务器端的一种方式， 一般来说，目前讨论区或部落格等，大多使用可以支持较多数据的 POST 方式上传啦！ 关于 GET 与 POST 的相关信息我们会在第二十章 WWW 服务器当中再次的提及！

5.4.2 文字接口下载器： wget

如果说 links 是在进行网页的『浏览』，那么 wget 就是在进行『网页数据的取得』。举例来说，我们的 Linux 核心是放置在 www.kernel.org 内，主要同时提供 ftp 与 http 来下载。我们知道可以使用 lftp 来下载数据，但如果想要用浏览器来下载呢？那就利用 wget 吧！

```
[root@www ~]# wget [option] [网址]
选项与参数：
若想要联机的网站有提供账号与密码的保护时，可以利用这两个参数来输入喔！
--http-user=usrname
--http-password=password
--quiet ：不要显示 wget 在抓取数据时候的显示讯息
更多的参数请自行参考 man wget 吧！ ^_^

# 范例一：请下载 2.6.39 版的核心
[root@www ~]# wget  \
&gt; http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.39.tar.bz2
--2011-07-18 16:58:26--  http://www.kernel.org/pub/linux/kernel/v2.6/..
Resolving www.kernel.org... 130.239.17.5, 149.20.4.69, 149.20.20.133, ...
Connecting to www.kernel.org&#124;130.239.17.5&#124;:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 76096559 (73M) [application/x-bzip2]
Saving to: `linux-2.6.39.tar.bz2'

88% [================================&gt;        ] 67,520,536  1.85M/s  eta 7s
```

你瞧瞧～很可爱吧！不必透过浏览器，只要知道网址后，立即可以进行档案的下载， 又快速又方便，还可以透过 proxy 的帮助来下载呢！透过修改 /etc/wgetrc 来设定你的代理服务器：

```
[root@www ~]# vim /etc/wgetrc
#http_proxy = http://proxy.yoyodyne.com:18023/  &lt;==找到底下这几行，大约在 78 行
#ftp_proxy = http://proxy.yoyodyne.com:18023/
#use_proxy = on

# 将他改成类似底下的模样，记得，你必须要有可接受的 proxy 主机才行！
http_proxy = http://proxy.ksu.edu.tw:3128/
use_proxy = on
```

5.5 封包撷取功能

很多时候由于我们的网络联机出现问题，使用类似 ping 的软件功能却又无法找出问题点，最常见的是因为路由与 IP 转递后所产生的一些困扰 (请参考防火墙与 NAT 主机部分)，这个时候要怎么办？最简单的方法就是『分析封包的流向』啰！透过分析封包的流向，我们可以了解一条联机应该是如何进行双向的联机的动作， 也就会清楚的了解到可能发生的问题所在了！底下我们就来谈一谈这个 tcpdump 与图形接口的封包分析软件吧！

5.5.1 文字接口封包撷取器： tcpdump

说实在的，对于 tcpdump 这个软件来说，你甚至可以说这个软件其实就是个黑客软件， 因为他不但可以分析封包的流向，连封包的内容也可以进行『监听』， 如果你使用的传输数据是明码的话，不得了，在 router 或 hub 上面就可能被人家监听走了！ 我们在第二章谈到的 CSMA/CD 流程中，不是说过有所谓的『监听软件』吗？这个 tcpdump 就是啦！ 很可怕吶！所以，我们也要来了解一下这个软件啊！(注：这个 tcpdump 必须使用 root 的身份执行)

```
[root@www ~]# tcpdump [-AennqX] [-i 接口] [-w 储存档名] [-c 次数] \
                       [-r 档案] [所欲撷取的封包数据格式]
选项与参数：
-A ：封包的内容以 ASCII 显示，通常用来捉取 WWW 的网页封包资料。
-e ：使用资料连接层 (OSI 第二层) 的 MAC 封包数据来显示；
-nn：直接以 IP 及 port number 显示，而非主机名与服务名称
-q ：仅列出较为简短的封包信息，每一行的内容比较精简
-X ：可以列出十六进制 (hex) 以及 ASCII 的封包内容，对于监听封包内容很有用
-i ：后面接要『监听』的网络接口，例如 eth0, lo, ppp0 等等的界面；
-w ：如果你要将监听所得的封包数据储存下来，用这个参数就对了！后面接档名
-r ：从后面接的档案将封包数据读出来。那个『档案』是已经存在的档案，
     并且这个『档案』是由 -w 所制作出来的。
-c ：监听的封包数，如果没有这个参数， tcpdump 会持续不断的监听，
     直到使用者输入 [ctrl]-c 为止。
所欲撷取的封包数据格式：我们可以专门针对某些通讯协议或者是 IP 来源进行封包撷取，
     那就可以简化输出的结果，并取得最有用的信息。常见的表示方法有：
     'host foo', 'host 127.0.0.1' ：针对单部主机来进行封包撷取
     'net 192.168' ：针对某个网域来进行封包的撷取；
     'src host 127.0.0.1' 'dst net 192.168'：同时加上来源(src)或目标(dst)限制
     'tcp port 21'：还可以针对通讯协议侦测，如 tcp, udp, arp, ether 等
     还可以利用 and 与 or 来进行封包数据的整合显示呢！

# 范例一：以 IP 与 port number 捉下 eth0 这个网络卡上的封包，持续 3 秒
[root@www ~]# tcpdump -i eth0 -nn
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
17:01:47.360523 IP 192.168.1.101.1937 &gt; 192.168.1.100.22: Flags [.], ack 196, win 6521
&lt;u&gt;17:01:47.362139 IP 192.168.1.100.22 &gt; 192.168.1.101.1937: Flags [P.], seq 196
17:01:47.363201 IP 192.168.1.100.22 &gt; 192.168.1.101.1937: Flags [P.], seq 472:636, ack
17:01:47.363328 IP 192.168.1.101.1937 &gt; 192.168.1.100.22: Flags [.], ack 636, win 6477
&lt;==按下 [ctrl]-c 之后结束
6680 packets captured              &lt;==捉下来的封包数量
14250 packets received by filter   &lt;==由过滤所得的总封包数量
7512 packets dropped by kernel     &lt;==被核心所丢弃的封包
```

如果你是第一次看 tcpdump 的 man page 时，肯定一个头两个大，因为 tcpdump 几乎都是分析封包的表头数据，用户如果没有简易的网络封包基础，要看懂粉难呐！ 所以，至少你得要回到网络基础里面去将 TCP 封包的表头资料理解理解才好啊！ ^_^！至于那个范例一所产生的输出范例中，我们可以约略区分为数个字段， 我们以范例一当中那个特殊字体行来说明一下：

- 17:01:47.362139：这个是此封包被撷取的时间，『时:分:秒』的单位；
- IP：透过的通讯协议是 IP ；
- 192.168.1.100.22 > ：传送端是 192.168.1.100 这个 IP，而传送的 port number 为 22，你必须要了解的是，那个大于 (>) 的符号指的是封包的传输方向喔！
- 192.168.1.101.1937：接收端的 IP 是 192.168.1.101， 且该主机开启 port 1937 来接收；
- [P.], seq 196:472：这个封包带有 PUSH 的数据传输标志， 且传输的数据为整体数据的 196~472 byte；
- ack 1：ACK 的相关资料。

最简单的说法，就是该封包是由 192.168.1.100 传到 192.168.1.101，透过的 port 是由 22 到 1937 ， 使用的是 PUSH 的旗标，而不是 SYN 之类的主动联机标志。呵呵！不容易看的懂吧！所以说，上头才讲请务必到 TCP 表头数据的部分去瞧一瞧的啊！

再来，一个网络状态很忙的主机上面，你想要取得某部主机对你联机的封包数据而已时， 使用 tcpdump 配合管线命令与正规表示法也可以，不过，毕竟不好捉取！ 我们可以透过 tcpdump 的表示法功能，就能够轻易的将所需要的数据独立的取出来。 在上面的范例一当中，我们仅针对 eth0 做监听，所以整个 eth0 接口上面的数据都会被显示到屏幕上， 不好分析啊！那么我们可以简化吗？例如只取出 port 21 的联机封包，可以这样做：

```
[root@www ~]# tcpdump -i eth0 -nn port 21
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
01:54:37.96 IP 192.168.1.101.1240 &gt; 192.168.1.100.21: . ack 1 win 65535
01:54:37.96 IP 192.168.1.100.21 &gt; 192.168.1.101.1240: P 1:21(20) ack 1 win 5840
01:54:38.12 IP 192.168.1.101.1240 &gt; 192.168.1.100.21: . ack 21 win 65515
01:54:42.79 IP 192.168.1.101.1240 &gt; 192.168.1.100.21: P 1:17(16) ack 21 win 65515
01:54:42.79 IP 192.168.1.100.21 &gt; 192.168.1.101.1240: . ack 17 win 5840
01:54:42.79 IP 192.168.1.100.21 &gt; 192.168.1.101.1240: P 21:55(34) ack 17 win 5840
```

瞧！这样就仅提出 port 21 的信息而已，且仔细看的话，你会发现封包的传递都是双向的，client 端发出『要求』而 server 端则予以『响应』，所以，当然是有去有回啊！ 而我们也就可以经过这个封包的流向来了解到封包运作的过程。举例来说：

1. 我们先在一个终端机窗口输入『 tcpdump -i lo -nn 』 的监听，
2. 再另开一个终端机窗口来对本机 (127.0.0.1) 登入『ssh localhost』

那么输出的结果会是如何？

```
[root@www ~]# tcpdump -i lo -nn
 1 tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
 2 listening on lo, link-type EN10MB (Ethernet), capture size 96 bytes
 3 11:02:54.253777 IP 127.0.0.1.32936 &gt; 127.0.0.1.22: S 933696132:933696132(0)
   win 32767 &lt;mss 16396,sackOK,timestamp 236681316 0,nop,wscale 2&gt;
 4 11:02:54.253831 IP 127.0.0.1.22 &gt; 127.0.0.1.32936: S 920046702:920046702(0)
   ack 933696133 win 32767 &lt;mss 16396,sackOK,timestamp 236681316 236681316,nop,
   wscale 2&gt;
 5 11:02:54.253871 IP 127.0.0.1.32936 &gt; 127.0.0.1.22: . ack 1 win 8192 &lt;nop,
   nop,timestamp 236681316 236681316&gt;
 6 11:02:54.272124 IP 127.0.0.1.22 &gt; 127.0.0.1.32936: P 1:23(22) ack 1 win 8192
   &lt;nop,nop,timestamp 236681334 236681316&gt;
 7 11:02:54.272375 IP 127.0.0.1.32936 &gt; 127.0.0.1.22: . ack 23 win 8192 &lt;nop,
   nop,timestamp 236681334 236681334&gt;
```

上表显示的头两行是 tcpdump 的基本说明，然后：

- 第 3 行显示的是『来自 client 端，带有 SYN 主动联机的封包』，
- 第 4 行显示的是『来自 server 端，除了响应 client 端之外(ACK)，还带有 SYN 主动联机的标志；
- 第 5 行则显示 client 端响应 server 确定联机建立 (ACK)
- 第 6 行以后则开始进入数据传输的步骤。

从第 3-5 行的流程来看，熟不熟悉啊？没错！那就是三向交握的基础流程啦！够有趣吧！ 不过 tcpdump 之所以被称为黑客软件之一可不止上头介绍的功能呐！ 上面介绍的功能可以用来作为我们主机的封包联机与传输的流程分析， 这将有助于我们了解到封包的运作，同时了解到主机的防火墙设定规则是否有需要修订的地方。

更神奇的使用要来啦！如果我们使用 tcpdump 在 router 上面监听『明码』的传输数据时， 例如 FTP 传输协议，你觉得会发生什么问题呢？ 我们先在主机端下达『 tcpdump -i lo port 21 -nn -X 』然后再以 ftp 登入本机，并输入账号与密码， 结果你就可以发现如下的状况：

```
[root@www ~]# tcpdump -i lo -nn -X 'port 21'
    0x0000:  4500 0048 2a28 4000 4006 1286 7f00 0001  E..H*(@.@.......
    0x0010:  7f00 0001 0015 80ab 8355 2149 835c d825  .........U!I.\.%
    0x0020:  8018 2000 fe3c 0000 0101 080a 0e2e 0b67  .....&lt;.........g
    0x0030:  0e2e 0b61 3232 3020 2876 7346 5450 6420  ...a220.(vsFTPd.
    0x0040:  322e 302e 3129 0d0a                      2.0.1)..

    0x0000:  4510 0041 d34b 4000 4006 6959 7f00 0001  E..A.K@.@.iY....
    0x0010:  7f00 0001 80ab 0015 835c d825 8355 215d  .........\.%.U!]
    0x0020:  8018 2000 fe35 0000 0101 080a 0e2e 1b37  .....5.........7
    0x0030:  0e2e 0b67 5553 4552 2064 6d74 7361 690d  ...gUSER.dmtsai.
    0x0040:  0a                                       .

    0x0000:  4510 004a d34f 4000 4006 694c 7f00 0001  E..J.O@.@.iL....
    0x0010:  7f00 0001 80ab 0015 835c d832 8355 217f  .........\.2.U!.
    0x0020:  8018 2000 fe3e 0000 0101 080a 0e2e 3227  .....&gt;........2'
    0x0030:  0e2e 1b38 5041 5353 206d 7970 6173 7377  ...8PASS.mypassw
    0x0040:  6f72 6469 7379 6f75 0d0a                 ordisyou..
```

上面的输出结果已经被简化过了，你必须要自行在你的输出结果当中搜寻相关的字符串才行。 从上面输出结果的特殊字体中，我们可以发现『该 FTP 软件使用的是 vsftpd ，并且使用者输入 dmtsai 这个账号名称，且密码是 mypasswordisyou』 嘿嘿！你说可不可怕啊！如果

使用的是明码的方式来传输你的网络数据？ 所以我们才常常在讲啊，网络是很不安全滴！

另外你得了解，为了让网络接口可以让 tcpdump 监听，所以执行 tcpdump 时网络接口会启动在 『错乱模式 (promiscuous)』，所以你会在 /var/log/messages 里面看到很多的警告讯息，通知你说你的网络卡被设定成为错乱模式！别担心，那是正常的。至于更多的应用，请参考 man tcpdump 啰！

例题：如何使用 tcpdump 监听 (1)来自 eth0 适配卡且 (2)通讯协议为 port 22 ，(3)封包来源为 192.168.1.101 的封包资料？答：tcpdump -i eth0 -nn 'port 22 and src host 192.168.1.101'

5.5.2 图形接口封包撷取器： wireshark

tcpdump 是文字接口的封包撷取器，那么有没有图形接口的？有啊！那就是 wireshark (注1) 这套软件。这套软件早期称为 ethereal ，目前同时提供文字接口的 tethereal 以及图形接口的 wireshark 两个咚咚。由于我们当初安装时预设并没有装这套，因此妳必须要先使用 yum 去网络安装喔！也可以拿出光盘来安装啦！有两套需要安装，分别是文字接口的 wireshark 以及图形接口的 wireshark-gnome 软件。安装方式如下：

```
[root@www ~]# yum install wireshark wireshark-gnome
```

启动这套软件的方法很简单，你必须要在 X Window 底下，透过『应用程序』-->『因特网』-->『wireshark network analyzer』就可以启动啦！启动的画面如下所示：

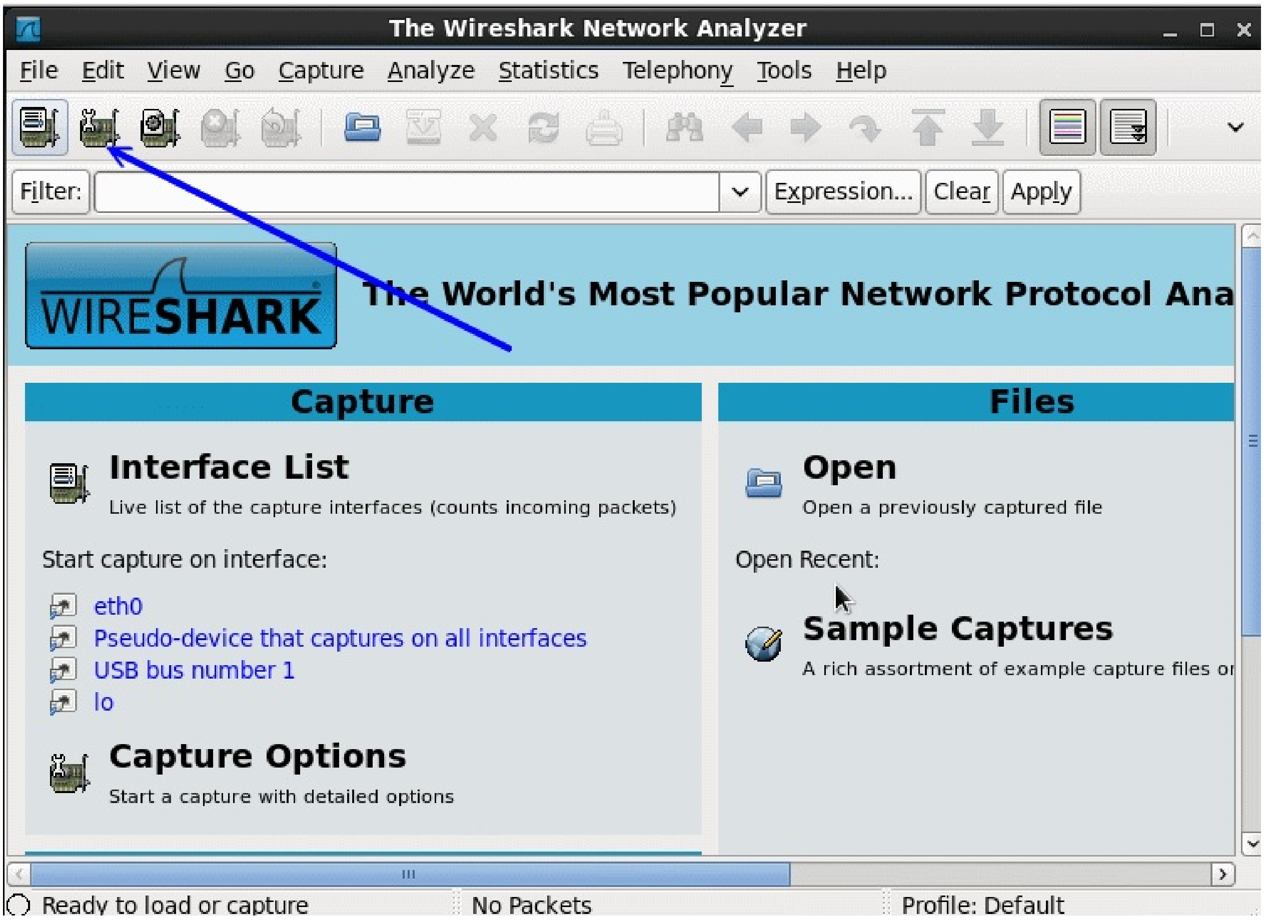


图 5.5-1、wireshark 的使用示意图

其实这一套软件功能非常强大！鸟哥这里仅讲简单的用法，若有特殊需求，就得要自己找找数据啰。 想要开始撷取封包前，得要设定一下监听的接口之类的，因此点选图 5.5-1 画面中的网络卡小图标吧！ 就会出现如下的画面给你选择了。

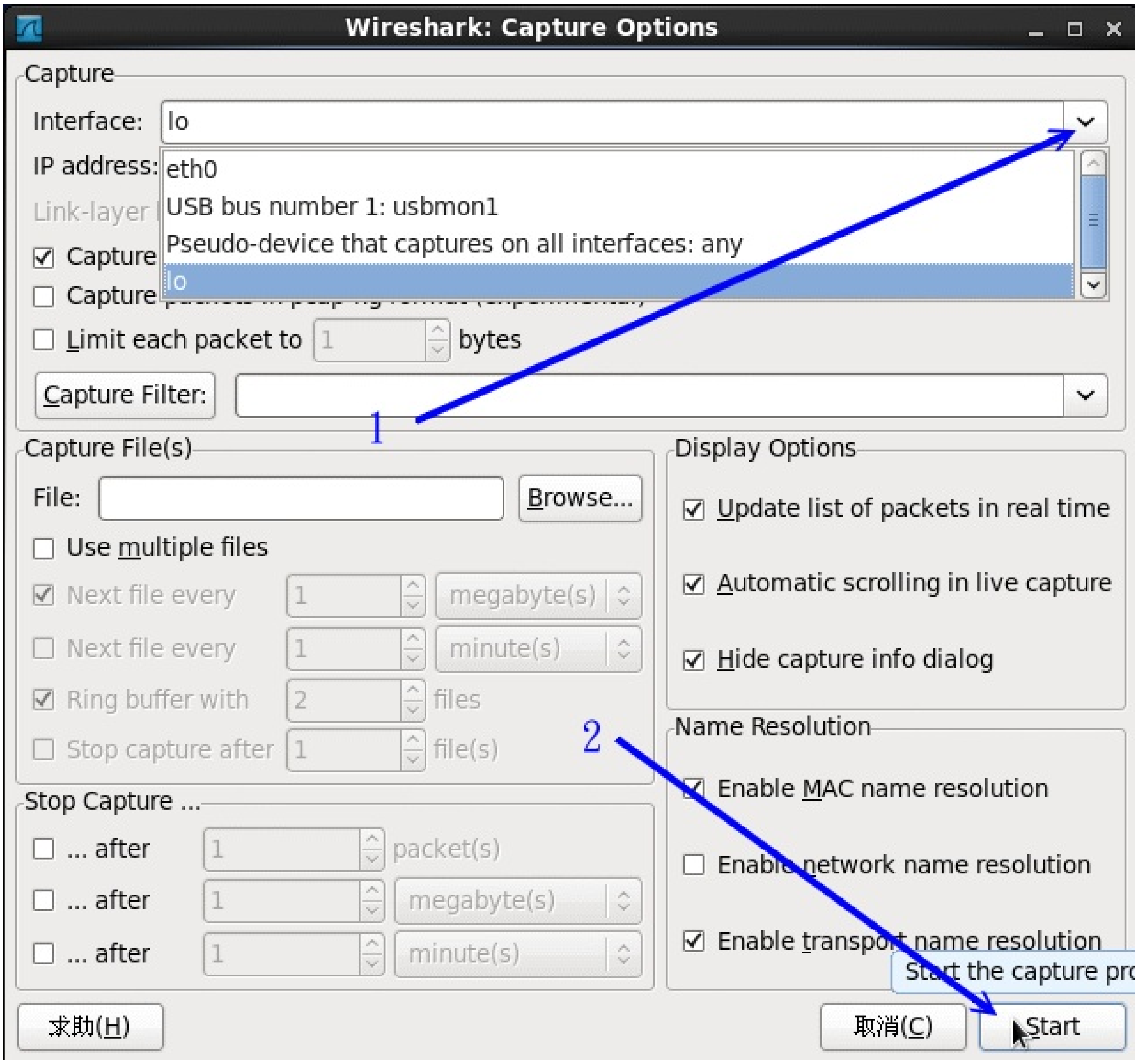


图 5.5-2、wireshark 的使用示意图

在上图中，你得先选择想要监听的接口，鸟哥这里因为担心外部的封包太多导致画面很乱，因此这里使用内部的 lo 接口来作为范例。你得要注意， lo 平时是很安静的！所以，鸟哥在点选了『start』之后，还有打开终端机， 之后使用『 ssh localhost 』来尝试登入自己，这样才能够获得封包喔！如下图所示：

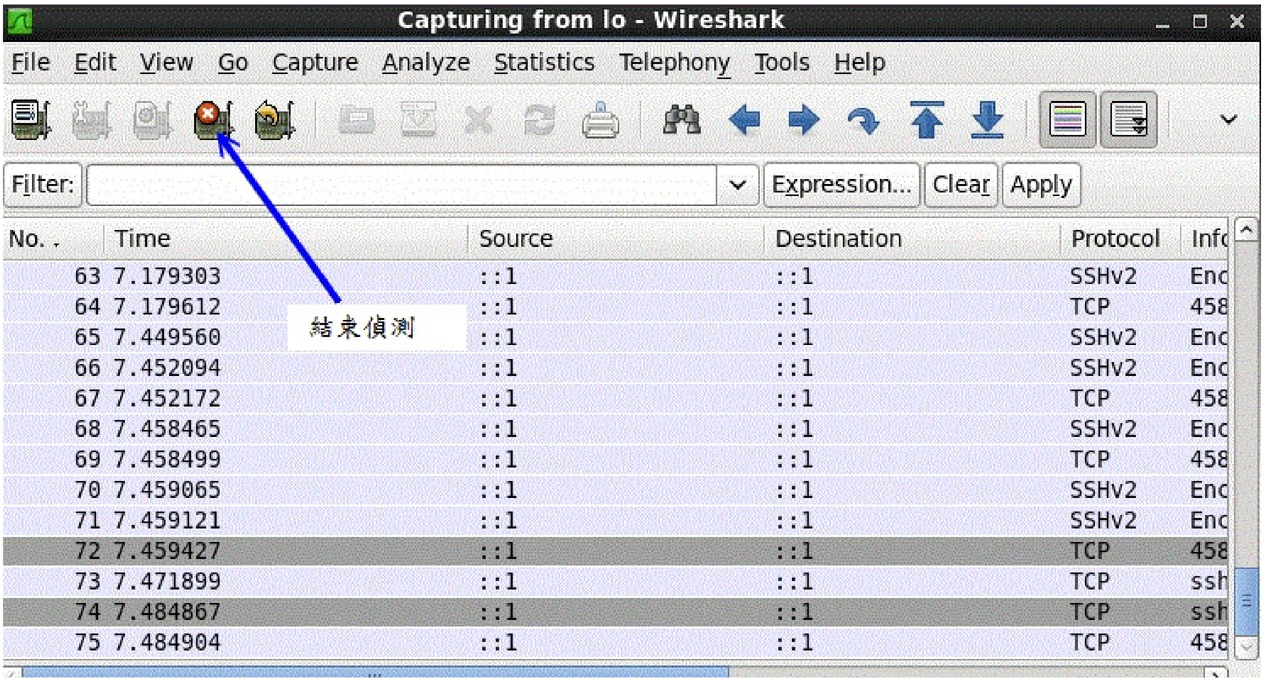


图 5.5-3、wireshark 的使用示意图

若没有问题，等到你撷取了足够的封包想要进行分析之后，按下图 5.5-3 画面中的停止小图示，那么封包撷取的动作就会终止， 接下来，就让我们来开始分析一下封包吧！

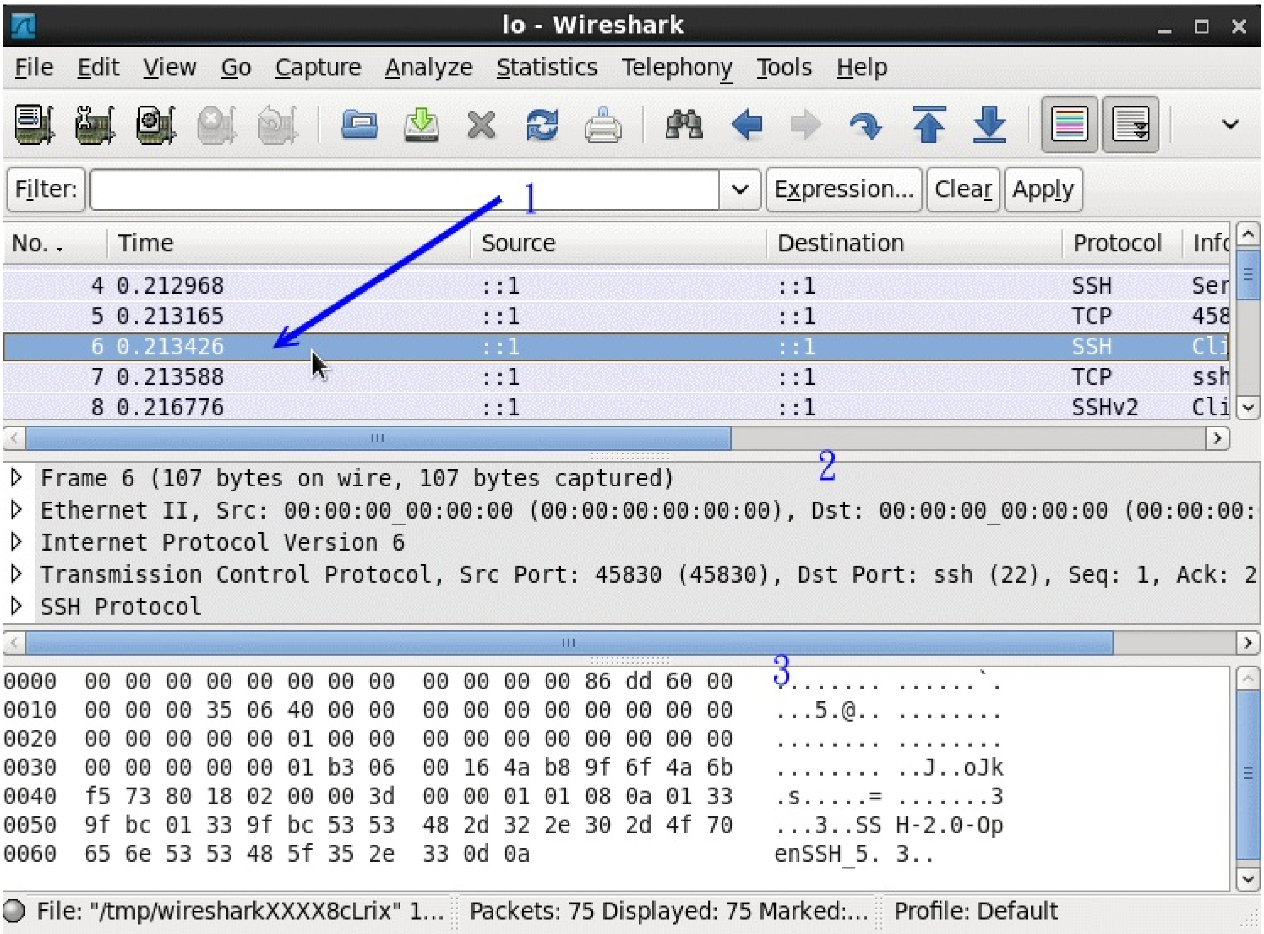


图 5.5-4、wireshark 的使用示意图

整个分析的画面如上所示，画面总共分为三大区块，你可以将鼠标光标移动到每个区块中间的移动棒， 就可以调整每个区块的范围大小了。第一区块主要显示的是封包的标头资料，内容就有点类似 tcpdump 的显示结果，第二区块则是详细的表头资料，包括讯框的内容、通讯协议的内容以及 socket pair 等等信息。 第三区块则是 16 进位与 ASCII 码的显示结果 (详细的封包内容)。

如果你觉得某个封包有问题，在画面 1 的地方点选该封包 (图例中是第 6 个封包)，那么画面 2 与 3 就会跟着变动！由于鸟哥测试的封包是加密数据的封包，因此画面 2 显示出封包表头，但画面 3 的封包内容就是乱码啦！ 透过这个 wireshark 你就可以一口气得到所需要的所有封包内容啦！而且还是图形接口的，很方便吧！

5.5.3 任意启动 TCP/UDP 封包的埠口联机： nc, netcat

这个 nc 指令可以用来作为某些服务的检测，因为他可以连接到某个 port 来进行沟通，此外，还可以自行启动一个 port 来倾听其他用户的联机呐！非常的不错用！如果在编译 nc 软件的时候给予『GAPING_SECURITY_HOLE』参数的话，嘿嘿！ 这个软件还可以用来取得客户端的 bash 哩！可怕吧！我们的 CentOS 预设并没有给予上面的参数， 所以我们不能够用来作为黑客软件～但是 nc 用来取代 telnet 也是个很棒的功能了！(有的系统将执行文件 nc 改名为 netcat 啦！)

```
[root@www ~]# nc [-u] [IP&#124;host] [port]
[root@www ~]# nc -l [IP&#124;host] [port]
选项与参数：
-l ：作为监听之用，亦即开启一个 port 来监听用户的联机；
-u ：不使用 TCP 而是使用 UDP 作为联机的封包状态

# 范例一：与 telnet 类似，连接本地端的 port 25 查阅相关讯息
[root@www ~]# yum install nc
[root@www ~]# nc localhost 25
```

这个最简单的功能与 telnet 几乎一样吧！可以去检查某个服务啦！不过，更神奇的在后面，我们可以建立两个联机来传讯喔！举个例子来说，我们先在服务器端启动一个 port 来进行倾听：

```
# 范例二：激活一个 port 20000 来监听使用者的联机要求
[root@www ~]# nc -l localhost 20000 &
[root@www ~]# netstat -tlunp &#124; grep nc
tcp        0      0 ::1:20000        :::*     LISTEN      5433/nc
# 启动一个 port 20000  在本机上！
```

接下来你再开另外一个终端机来看看，也利用 nc 来联机服务器，并且输入一些指令看看喔！

```
[root@www ~]# nc localhost 20000
   &lt;==这里可以开始输入字符串了！
```

此时，在客户端我们可以打入一些字，你会发现在服务器端会同时出现你输入的字眼呐！ 如果你同时给予一些额外的参数，例如利用标准输入与输出 (stdout, stdin) 的话，那么就可以透过这个联机来作很多事情了！ 当然 nc 的功能不只如此，你还可以发现很多的用途喔！ 请自行到你主机内的 /usr/share/doc/nc-1.84/scripts/ 目录下看看这些 script ，有帮助的呐！ 不过，如果你需要额外的编译出含有 GAPING_SECURITY_HOLE 功能， 以使两端联机可以进行额外指令的执行时，就得要自行下载原始码来编译了！

5.6 重点回顾

- 修改网络接口的硬件相关参数，可以使用 ifconfig 这个指令，包括 MTU 等等；
- ifup 与 ifdown 其实只是 script ，在使用时，会主动去 /etc/sysconfig/network-scripts 下找到相对应的装置配置文件，才能够正确的启动与关闭；
- 路由的修改与查阅可以使用 route 来查询，此外， route 亦可进行新增、删除路由的工作；
- ip 指令可以用来作为整个网络环境的设定，利用 ip link 可以修改『网络装置的硬件相关功能』， 包括 MTU 与 MAC 等等，可以使用 ip address 修改 TCP/IP 方面的参数，包括 IP 以及网域参数等等， ip route 则可以修改路由！
- ping 主要是透过 ICMP 封包来进行网络环境的检测工作，并且可以使用 ping 来查询整体网域可接受最大的 MTU 值；
- 侦察每个节点的联机状况，可以使用 traceroute 这个指令来追踪！
- netstat 除了可以观察本机的启动接口外，还可以观察 Unix socket 的传统插槽接口数据；
- host 与 nslookup 预设都是透过 /etc/resolv.conf 内设定的 DNS 主机来进行主机名与 IP 的查询；
- lftp 可以用来匿名登录远程的 FTP 主机；
- links 主要的功能是『浏览』，包括本机上 HTML 语法的档案， wget 则主要在用来下载 WWW 的资料；
- 撷取封包以分析封包的流向，可使用 tcpdump ，至于图形接口的 wireshark 则可以进行更为详细的解析。
- 透过 tcpdump 分析三向交握，以及分析明码传输的数据，可发现网络加密的重要性。
- nc 可用来取代 telnet 进行某些服务埠口的检测工作。

5.7 本章习题

- 暂时将你的 eth0 这张网络卡的 IP 设定为 192.168.1.100 ，如何进行？ifconfig eth0 192.168.1.100
- 我要增加一个路由规则，以 eth0 连接 192.168.100.100/24 这个网域，应该如何下达指令？route add -net 192.l68.100.0 netmask 255.255.255.0 dev eth0
- 我的网络停顿的很厉害，尤其是连接到 tw.yahoo.com 的时候，那么我应该如何检查那个环节出了问题？traceroute tw.yahoo.com
- 我发现我的 Linux 主机上面有个联机很怪异，想要将他断线，应该如何进行？以 root 的身份进行『netstat -anp |more』查出该联机的 PID，然后以『 kill -9 PID 』踢掉该联机。
- 你如何知道 green.ev.ncku.edu.tw 这部主机的 IP ？方法很多，可以利用 host green.ev.ncku.edu.tw 或 dig green.ev.ncku.edu.tw 或 nslookup green.ev.ncku.edu.tw 等方法找出
- 请找出你的机器上面最适当的 MTU 应该是多少？请利用『ping -c 3 -M do -s MTU yourIP 』找出你的 IP 的 MTU 数值。 事实上，你还可以先以 ip 设定网络卡较大的 MTU 后，在进行上述的动作，才能够找出网域内适合的 MTU。
- 如何在终端机接口上面进行 WWW 浏览？又该如何下载 WWW 上面提供的档案？要浏览可以使用 links 或 lynx ，至于要下载则使用 wget 这个软件。
- 在终端机接口中，如何连接 bbs.sayya.org 这个 BBS ？利用 telnet bbs.sayya.org 即可连接上
- 请自行以 tcpdump 观察本机端的 ssh 联机时，三向交握的内容
- 请自行回答：为何使用明码传输的网络联机数据较为危险？并自行以软件将封包取出，并与同学讨论封包的信息
- 请自行至 Internet 下载 nc(netcat) 的原始码，并且编译成为具有 GAPING_SECURITY_HOLE 的参数， 然后建立一条联机使用 -e /bin/bash 尝试将本地端的 bash 丢给目的端执行 (特殊功能，可让 client 取得来自主机的 bash)。

5.8 参考数据与延伸阅读

- 注1：wireshark 的官网网址：http://www.wireshark.org/

2002/07/31：第一次完成日期！ 2003/08/19：重新编排版面，加入 jmcce 的安装以及 MTU 的相关说明 2003/08/20：加入课后练习去 2003/09/19：加入参考用解答咯！ 2005/03/24：route 的指令参数写错了！已经订正！ 2006/07/24：将旧的文章移动到 此处 2006/07/24：拿掉相关性不高的 JMCCE 中文终端机； 将 Windows 系统的 MTU 检测修改方法移除。 也拿掉 ncftp 的说明 2006/08/02：修改了很多部分，加入一些封包侦测的功能程序，tcpdump, nc 等指令！ 2010/08/28：将旧的，基于 CentOS 4.x 所撰写的文章放置于此处 2010/09/03：加入 links 取消 lynx，ethereal 改成 wireshark，gaim 改成 pidgin 了，nc 指令的用法跟前几版有点不同。 2011/07/18：将基于 CentOS 5.x 的文章移动到此处 2011/07/18：将资料修订为 CentOS 6.x 的模样！不过 tcpdump 的变化不大，部分数据为 CentOS 5.x 的撷取示意！

第六章、 Linux 网络侦错

最近更新日期：2011/07/19

虽然我们在第四章谈完了连上 Internet 的方法，也大略介绍了五个主要的网络检查步骤。不过，网络是很复杂的东西， 鸟哥也是接触了 Linux 这么多年之后才对网络与通讯协议有点认识而已，要说到熟悉与了解，那还有段路要走哩。 总之，为了让大家对网络问题的解决有个方向可以进行处理，鸟哥底下列出一些常见的问题，希望对大家有点帮助！

6.1 无法联机原因分析

老是看到有朋友在网络上哀嚎说：『我的网络不通啊！』还有比较奇怪的是『啊！怎么网络时通时不通』之类的问题， 这类的问题其实主要可以归类为硬件问题与软件设定问题。硬件的问题比较麻烦，因为需要透过一些专门的装置来分析硬件； 至于软件方面，绝大部分都是设定错误或者是观念错误而已，比较好处理啦 (第四章谈到的就是软件问题)！ OK！我们先来看看网络在哪里可能会出问题吧！

6.1.1 硬件问题：网络线材、网络设备、网络布线等

在第二章的网络基础当中我们曾提到很多的网络基础概念， 以及一些简单的硬件维护问题。以一个简单的星形联机来说，我们可以假设他的架构如同下图所示：

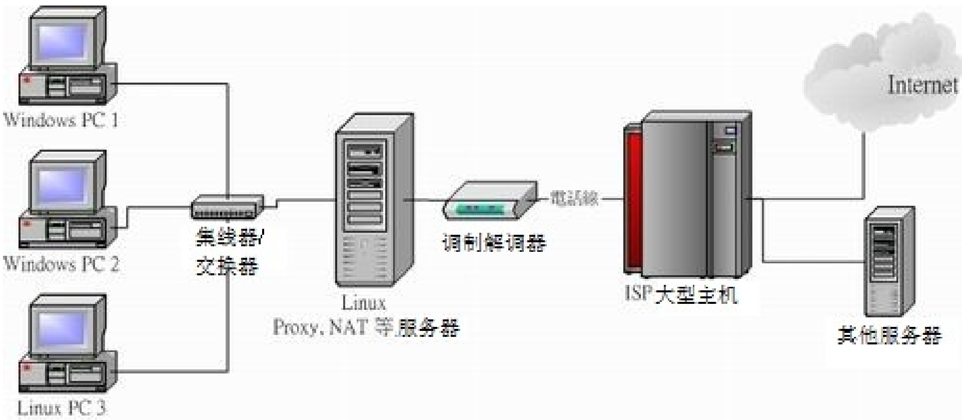


图 6.1-1、局域网络的联机状态示意图

在上面的图示当中，"Linux PC3" 要连到 Internet 上面去的话，需要透过网络线、交换器、NAT 主机 (Linux 服务器或 IP 分享器)、ADSL 调制解调器，附挂电话线路、ISP 自己的机房交换器，以及 Internet 上面的所有媒体设备 (包括路由器、网桥、其他网络线等等)；那么哪些地方可能会出问题啊？

1. 网络线材的问题：

   在上面的图示中，可以发现，其实网络接口设备中，使用最多的就是网络线啦！要注意网络线分成并行线与跳线 (RJ-45接头)，而并不是所有的设备都支持自动分辨跳线与并行线的功能的！ 所以你必须要了解到你的设备 (Hub/Switch/调制解调器) 所支持的网络线；另外，如果你的网络线有经过门缝处或者是容易凹折处， 那很有可能由于经常性的凹折导致电子讯号不良，所以你需要注意一下这些事情：

   - 网络线被截断；
   - 网络线过度扭曲变形造成讯号不良；
   - 自制网络接头 (如 RJ-45 跳线头) 品质不良；

    - 网络接头与设备 (如 Hub) 接触不良；
2. 网络卡、Hub 及 Router 等网络设备的问题：

   另外，还有一些网络设备也会有问题，常见的问题如下：

    - 网络卡不稳定、质量不佳，或者与整体系统的兼容度不佳 (网络卡也是会坏的)；
    - 各网络设备的接头不佳，接触不良，造成讯号衰减 (经常的拔插就有可能发生)；
    - 由于网络设备所在环境恶劣 (例如过热) 导致的当机问题 (鸟哥经常遭遇到 switch 热当的问题)；
    - 各网络设备使用方法不良，造成设备功能衰减 (switch 常常插电/断电容易坏喔)；
3. 设备配置的规则：

   在各个设备的配置上是有一定的规则的，而最容易发生的问题就是太长的网络线会造成讯号的衰减， 导致网络联机的时间太长甚至无法联机。我们曾在网络基础当中谈过以太网络最长的支持距离 (10BASE5 最长可达 500m)， 还有一些其他网络媒体配置的问题你必须晓得的：

    - 使用错误的网络线，最常发生在并行线与跳线的分别 (现在比较少见这个问题了)！
    - 架设的网络线过长，导致讯号衰减太严重。例如以太网络 CAT5e 的线理论限制长度大概是在 90 公尺左右 (虽然 10BASE5 可达 500m)，若两个设备 (Hub/主机之间) 长度大于 90 公尺时，自然就容易出现讯号发生问题了！
    - 其他噪声的干扰，最常发生在网络线或者网络设备旁边有太强的磁波；
    - 局域网络上面，节点或者其他的设备太多，过去我们常以所谓的 543 原则来说明：(注1)
        - 5 个网段 (segment)。所谓 segment 就在物理连接上最接近的一组计算机，在一个 BNC 网段里面最多只能接 30 台计算机，且网线总长不能超过 185m。
        - 4 个增益器 (repeater)。也就是将信号放大的装置。
        - 3 个计算机群体 (population)。这个不好理解，也就是说前面所说的 5 个 segment 之中， 只能有 3 个可以装计算机，其它两个不行。

上述是一些最常见的硬件问题，当然啦，有的时候是设备本身就有问题，而我们在网络基础里面谈到的那个很重要的『 网络布线 』的情况，也是造成网络停顿或通顺与否的重要原因呐！所以，硬件问题的判断比较困难点。 好～底下我们再来聊一聊软件设定的相关问题。

6.1.2 软件问题：IP 参数设定、路由设定、服务与防火墙设定等

所谓的软件问题，绝大部分就是 IP 参数设定错误啊，路由不对啊，还有 DNS 的 IP 设定错误等等的， 这些问题都是属于软件设定啦！只要将设定改一改，利用一些侦测软件查一查，就知道问题出在哪里了！基本的问题有：

1. 网络卡的 IP/netmask 设定错误： 例如：同一个 IP 在同一个网段中出现造成 IP 冲突、子网掩码设定错误、网络卡的驱动程序使用错误、网络卡的 IRQ、 I/O Address 的设定冲突等等；
2. 路由的问题 (route table)： 最常见的就是预设路由 (default gateway) 设定错误了！或者是路由接口不符所导致的问题， 使得数据封包没有办法顺利的送出去。
3. 通讯协议不相符： 最常发生在不同的操作系统之间的通讯传输，例如早期 Windows 98 与 Windows 2000 之间的『网芳』 若要达成沟通，则 Windows 98 必须要加装 NetBEUI 这个通讯协议才行。又例如两部 Linux 主机要透过 NFS 通讯协议传输数据时，两边都得要支持 rpcbind 这个启动 RPC 协议的程序才行！ 这些通讯协议我们都会在后面的章节分别介绍的啦！
4. 网络负荷的问题 (loading)： 当同时有大量的数据封包涌进 Server 或者是 Hub 或者是同一个网域中， 就有可能造成网络的停顿甚至挂点！另外，如果区网内有人使用 BT (P2P 软件) 或者是有人中毒导致蠕虫充满整个区网，也会造成网络的停顿问题；
5. 其他问题： 例如：一些 port 被防火墙挡住了，造成无法执行某些网络资源；应用程序本身的 Bug 问题；应用程序中用户的网络设定错误；以及不同的操作系统的兼容性问题等等。

6.1.3 问题的处理

既然问题发生了，就要去处理他啊！那如何处理呢？以上面的星形联机图示为例，把握两个原则：

- 先由自身的环境侦测起，可以由自身 PC 上的网络卡查起，到网络线、到 Hub 再到调制解调器等等的硬件先检查完。 在这个步骤当中，最好用的软件就是 ping ，而你最好能有两部以上的主机来进行联机的测试；
- 确定硬件没问题了，再来思考软件的设定问题！

实际上，如果网络不通时，你可以依序这样处理：

1. 了解问题：这个问题是刚刚发生？还是因为之前我做了什么动作而导致无法联机？ 例如之前鸟哥曾经更新过一个核心，结果该核心并不能驱动鸟哥的新网卡...
2. 确认 IP：先看看自己的网卡有无驱动？能否取得正确的 IP 相关参数来联机？
3. 确认区网联机：利用 ping 来沟通两部主机 (路由器或 IP 分享器)，确定网络线与中继的 hub/switch 工作正常；
4. 确认对外联机：看主机或 IP 分享器能否依据第四章的方法顺利取得 IP 参数，并以 ping 的方法确定对外联机是可以成功的 (例如 ping 168.95.1.1)；
5. 确认 DNS 查询：利用 nslookup 或 host 或 dig 检查 www.google.com 看看；
6. 确认 Internet 节点：可以利用 traceroute 检查各节点是否没问题？

7. 确认对方服务器正常服务：是否对方服务器忙线中？或他的机器挂了？
8. 确认我方服务器：如果是别人连不上我这部主机，那检查主机某些服务正确启动否？可利用 netstat 检查。或是否某些安全机制的软件没有设定好，例如 SELinux 这项机制；
9. 防火墙或权限的问题：是否由于权限设定错误所致？ 是否由于你的机器有防火墙忘记启用可联机的埠口所致？这个可以透过 tcpdump 来处理！

透过这些处理动作后，一般来说，应该都可以解决你无法上网的问题了！当然啦，如果是硬件的问题，那么鸟哥也无法帮你， 你可能最需要的是......『送修吧孩子！』

6.2 处理流程

既然知道上面已经谈到的几个小重点了，接下来当然是一个一个的给他处理掉啊！ 底下我们就得要一步一脚印的开始检查的流程啊！

6.2.1 步骤一：网络卡工作确认

其实，网络一出问题的时候，你应该从自己可以检查的地方检查起，因此， 最重要的地方就是检查你的网络卡是否有工作的问题啦！检查网络卡是否正常工作的方法如下：

1. 确定网络卡已经驱动成功： 如果网络卡没有驱动成功，其他的，免谈！所以你当然需要驱动你的网络卡才行！确认网络卡是否被驱动，可以利用 lspci 以及 dmesg 这两个咚咚来查询相关的设备与模块的对应。详情请参考：第四章的相关说明。 再次强调，捉不到网卡驱动程序，除了自己编译之外，再购买一张便宜的网卡来应付着用，是不错的想法！

2. 确定可以手动直接建立 IP 参数： 在顺利的加载网络卡的模块，并且『取得网络卡的代号』之后，我们可以利用 ifconfig 或 ip 来直接给予该网络卡一个网络地址试看看！ 看能否给予 IP 设定呢？例如：

   ```
   [root@www ~]# ifconfig eth0 192.168.1.100
   ```

   来直接建立该网络卡的 IP ，然后直接输入 ifconfig 看能否查阅到刚刚设定好的参数即可。 如果可以建立起该 IP ，就以 ping 来检测看看：

   ```
   [root@www ~]# ping 192.168.1.100
   ```

   如果有响应的话，那表示这个网卡的设定应该是没有问题了！再来则是开始检测一下局域网络内的各个连接硬件啦！

6.2.2 步骤二：局域网络内各项连接设备检测

在确认完了最重要的网络卡设定之后，并且确定网络卡是正常的之后，再接着下来则是局域网络内的网络连接情况了！假设你是按照 图 6.1-1 所设定的星形联机局域网络架构，那么你必须要知道整个『网域』的概念！

1. 关于网域的概念： 能否成功的架设出区网，与网域的概念有关！所以，你要知道所谓的 192.168.1.0/24 这种网域的表达方式所代表的意义， 且子网掩码 (Netmask) 的意义也得了解。如果忘记了，请回去第二章网络基础再翻一翻。

2. 关于 Gateway 与 DNS 的设定： Gateway 与 DNS 最容易被搞混～这两个并非是填写你的 Linux 主机的 IP 喔！应该是要填写 IP 分享器 (或 NAT 主机) 的 IP 在 Gateway 中，填写 168.95.1.1 在 DNS 的 IP 设定当中！不能够搞错啊！

3. 关于 Windows 端的工作组与计算机名称： 假如你还需要资源共享，那么你就必须在 windows 系统中开放档案分享， 并且建议所有的计算机将『工作组』设定相同，但『计算机名称』则不能相同！不过，这个只与网芳及 SAMBA 服务器有关。

假设你的区网内所有的主机 IP 都设定正确了，那么接下来你就可以使用 ping 来测试两部区网内主机的联机， 这个联机的动作可以让你测试两部主机间的各项设备，包括网络线、 Hub/Switch 等等的咚咚！ 如果无法测试成功，那就请了解一下：

1. IP 参数是否设定正确： 再次强调，先决定 IP/netmask 是对的！鸟哥在上课的时候常常发现同学无法连到我的主机上， 一经使用 ifconfig 才发现他们与我的 IP 不在同一个网段内，就是会有这样的情况发生啊！唉～

2. 联机的线材问题： 包括我们前面提到的网络线本身折损、过度缠绕造成的讯号衰减问题等等， 另外，有些比较旧的 Hub/Switch 或者是 ADSL 调制解调器，由于没有 Auto MDI/MDIX 的功能，所以无法自动的分辨跳线与否， 那么当你用错网络线的时候，也就无法接通啦！另外，早期我们常常会说， 最简单判断每部主机是否顺利连接到 Hub/Switch 可以透过连接到 Switch 上的灯号来判断， 不过，由于有时候网络线本身讯号不良，虽然灯号还是会亮，不过就是无法连接到 Switch 的情况 (鸟哥自己就曾发生过啊)，此时，跟朋友借一条 OK 的网络线来测试看看吧！

3. 网卡或 Hub/Switch 本身出问题： 有一次鸟哥无法在外部连接到鸟哥的主机，怀疑是挂点了，结果冲到主机所在办公室察看，咦！ 主机是好好的嘛！那怎么会无法联机呢？原因是...室内环境通风不良，加上 Switch 所在处温度过高， 加上那部旧的 switch 『刚好』风扇坏了，哈！就这样『 switch 当机 』 在重新启动 switch (拔掉再插上电源线) 后就正常了。所以啰，很多情况都是会发生的， 而局域网络内的环境也很容易影响到联机质量啊！

确定自己主机的 IP 与网卡没有问题，加上内部区网透过 ping 也测试过没有问题，接下来就是要『取得可以对外联机的 IP 参数』啦！这个重要！

6.2.3 步骤三：取得正确的 IP 参数

什么叫『取得正确的 IP 参数』啊？还记得我们谈过如果要顺利的连接上 Internet 的话， 必须要可以跟 public IP 进行沟通才行，而与 public IP 取得沟通的方法，在台湾比较常见的有 ADSL, Cable modem, 学术网络, 电话拨接等等。在 CentOS 当中，我们可以透过修改 /etc/sysconfig/network-scripts/ifcfg-eth0，或者是利用 rp-pppoe 来进行拨接， 无论如何，你就是得要连接到某个 ISP 去就是了～

在你确认所有的区网没有问题之后，参考一下第四章的介绍，连上之后，立即以 ifconfig 看看有没有捉到正确的 IP 啊？ 在台湾如果使用 ADSL 联机的话，你应该可以顺利的取得一组正确的 Public IP 参数的！

Tips: 曾有国外的华人朋友来信说到，他们使用 ADSL 拨接之后竟然取得一组 Private IP ， 害他们没有办法架站！他们想请问这样的情况是否合理。如果你熟悉路由相关的概念之后， 当然会知道：『这当然合理！』，因为你取得的 IP 只是为了要连接到 ISP 去而已， 而 ISP 与你的主机当然可以透过 Private IP 来联机啊！如果是这样的话，那么你就肯定无法架站了！ ^_^



另外，最常发现无法顺利取得 IP 的错误就是『BOOTPROTO』这个设定值设定错了！因为 static 与 dhcp 协议所产生的 IP 要求是不一样的啊！还记得吧！要特别留在 ifcfg-eth0 里面的设定参数喔！另外，如果你是使用 ADSL 拨接的， 但是老是无法拨接成功，那么建议你可以这样试看看：

- 将 ADSL 的调制解调器整个关机，将 Switch/Hub 也关掉电源；
- 静待十分钟，等这些设备比较『凉快』一点后，再重新插上电源；
- 将 Linux 连接到 ADSL 的那块网卡 (假设为 eth0) 在 ifcfg-eth0 内，『ONBOOT』设定为 no， 重新启动网络 (/etc/init.d/network restart)，然后再执行 adsl-start
- 如果还是无法拨接成功，并且你已经确认内部网域没有问题，那请中华电信的工程人员来帮忙你处理吧！

因为很多时候都是由于网络媒体过热，也有可能主机内部的一些网络参数有点问题， 所以，干脆就不要启动网卡，让 adsl-start 自动去启动网卡即可！ 如果顺利取得 IP 后，却还是无法顺利连到 Internet 上面时，你觉得还有哪些地方需要处理的呢？

Tips: 为了避免 switch 以及 ADSL 调制解调器热当，鸟哥以及一些重度网络用户的朋友，就买了桌上型小电扇，配合定时器 (timer) 来定时的向这两个设备吹风～为啥需要定时器？因为担心电风扇一直开着会烧掉...



6.2.4 步骤四：确认路由表的规则

如果你已经顺利取得正确的 IP 参数的话，那么接下来就是测试一下是否可以连上 Internet 啊！ 鸟哥建议你可以尝试使用 ping 来连连看 Hinet 的 DNS 主机，也就是 168.95.1.1 那部机器啦！

```
[root@www ~]# ping -c 3 168.95.1.1
```

如果有响应，那就表示你的网络『基本上已经没有问题，可以连到 Internet 了！』， 那如果没有响应呢？明明取得了正确的 IP 却无法连接到外部的主机，肯定有鬼！呵呵！没错！ 还记得我们在网域内资料的传输可以直接透过 MAC 来传送， 但如果不在区网内的数据，则需要透过路由，尤其是那个预设路由 (default route) 来帮忙转递封包吧！ 所以说，如果你的 public IP 无法连接到外部 (例如 168.95.1.1) ，可能的问题就出在路由与防火墙上面了。假设你没有启动防火墙，那问题就缩小到剩下路由啰～ 那路由的问题如何检查？就用 route -n 来检查啊！

例题：假设有个使用 ADSL 拨接的 Linux 主机，他的路由表如下，你觉得出了什么问题？

```
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
59.104.200.1    0.0.0.0         255.255.255.255 UH    0      0        0 ppp0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth0
127.0.0.0       0.0.0.0         255.0.0.0       U     0      0        0 lo
0.0.0.0         192.168.1.2     0.0.0.0         UG    0      0        0 eth0
```

答：仔细看到上面的路由输出，第一条是 ppp0 产生的 public IP 接口，第二条是 eth0 的内部网域接口，再看到最后一条的 0.0.0.0/0.0.0.0 这个预设路由，竟然是内部网域的 eth0 为 gateway ？不合理，最大的问题应该是出在 ifcfg-eth0 里面不小心设定了『GATEWAY=192.168.1.2』 所致，解决的方法为：

1. 取消 ifcfg-eth0 内 GATEWAY=192.168.1.2 那一行，(该行亦可能出现在 /etc/sysconfig/network 内)
2. 重新启动网络 /etc/init.d/network restart
3. 重新进行拨接： adsl-stop; adsl-start

另外一个可能发生的情况，就是：『忘记设定预设路由』啦！例如使用 ifconfig 手动重新设定过网络卡的 IP 之后，其实路由规则是会被更新的，所以预设路由可能就会不见了！那个时候你就得要利用 route add 来增加预设路由啰！

6.2.5 步骤五：主机名与 IP 查询的 DNS 错误

如果你发现可以 ping 到 168.95.1.1 这个 Internet 上面的主机，却无法使用浏览器在网址列浏览 http://www.google.com 的话，那肯定 99% 以上问题是来自于 DNS 解析的困扰！解决的方法就是直接到 /etc/resolv.conf 去看看设定值对不对啊！一般常见的内容是这样的：

```
[root@www ~]# vim /etc/resolv.conf
nameserver 168.95.1.1
nameserver 139.175.10.20
```

最常见的错误是『那个 nameserver 的拼字写错了！』真是最常见的问题～另外，如果 client 端是 Windows 系统呢？常常初学者会搞错的地方就是在 windows 的设定了！要注意：Windows 端的 DNS 设定与主机端 /etc/resolv.conf 的内容相同即可！很多初学者都以为 TCP/IP 内的 DNS 主机是填上自己的 Linux 主机，这是不对的 (除非你自己的 Linux 上面有 DNS 服务) ！你只要填上你的 ISP 给你的 DNS 主机 IP 位置就可以了

另外，每一部主机都会有主机名 (hostname) ，预设的主机名会是 localhost ，这个主机名会有一个 127.0.0.1 的 IP 对应在 /etc/hosts 当中。如果你曾经修改过你的主机名，该主机名却无法有一个正确 IP 的对应， 那么你的主机在开机时，可能会有好几十分钟的延迟。 所以啰，那个 /etc/hosts 与你的主机名对应，对于内部私有网域来说，是相当重要的设定项目呢！

6.2.6 步骤六：Linux 的 NAT 服务器或 IP 分享器出问题

NAT 服务器最简单的功能就是 IP 分享器啦！NAT 主机一定是部路由器，所以你必须要在 Linux 上面观察好正确的路由信息。否则肯定有问题。另外， NAT 主机上面的防火墙设定是否合理？ IP 分享器上面是否有设定抵挡的机制等等，都会影响到对外联机是否能够成功的问题点。 关于 NAT 与防火墙我们会在后续的章节继续介绍的啦！

6.2.7 步骤七：Internet 的问题

Internet 也会出问题喔！当然啦～没有任何东西是不会出问题的！举例来说，好几年前台湾西岸因为施工的关系， 导致南北网络骨干缆线被挖断，结果导致整个 Internet 流量的大塞车！这就是 Internet 的问题～还有，数年前 Study Area 网站放置的地点由于路由器设定出了点差错， 结果导致连接速度的缓慢。这都不是主机本身出问题，而是 Internet 上面某个节点出了状况。 想要确认是否问题来自 Internet 的话，就使用 traceroute 吧！ 查察看问题是来自那个地方再说！

6.2.8 步骤八：服务器的问题

如果上述的处理都 OK ，却无法登入某部主机时，我想，最大的问题就是出现在主机的设定啦！这包括有：

- 服务器并没有开放该项服务：例如主机关闭了 telnet ，那你使用 telnet 去联机，是无法连接上的啦！
- 主机的权限设定错误：例如你将某个目录设定为 drwx------ ，该目录拥有者为 root ， 你却将该目录开放给 WWW 来浏览，由于 WWW 无法进入该目录，所以当然无法正确的给客户端浏览啊！ 这是最典型的权限设定错误的情况啊！
- 安全机制设定错误：例如 SELinux 是用来更细微控管主机存取的一种核心机制，结果你启动的是系统原先不支持的类型， 那么 SELinux 反而可能会抵挡该服务的提供！而其他例如 /etc/hosts.deny, PAM 模块等等， 都可能造成使用者无法登入的问题！这就不是网络问题，而是主机造成联机无法成功！
- 防火墙问题：防火墙设定错误也是一个很常见的问题，你可以使用 tcpdump 来追踪封包的流向， 以顺利的了解防火墙是否设定错误。

基本上，一个网络环境的检测工作可不是三言两语就讲的完的～而且常常牵涉到很多经验的问题～ 请你常常到一些讲座的场合去听听看大家的经验，去 google 看看人家的解决方法，都有助于让你更轻易的解决网络问题的喔！ ^_^！鸟哥也将上述的动作规划成一个流程图，参考看看：

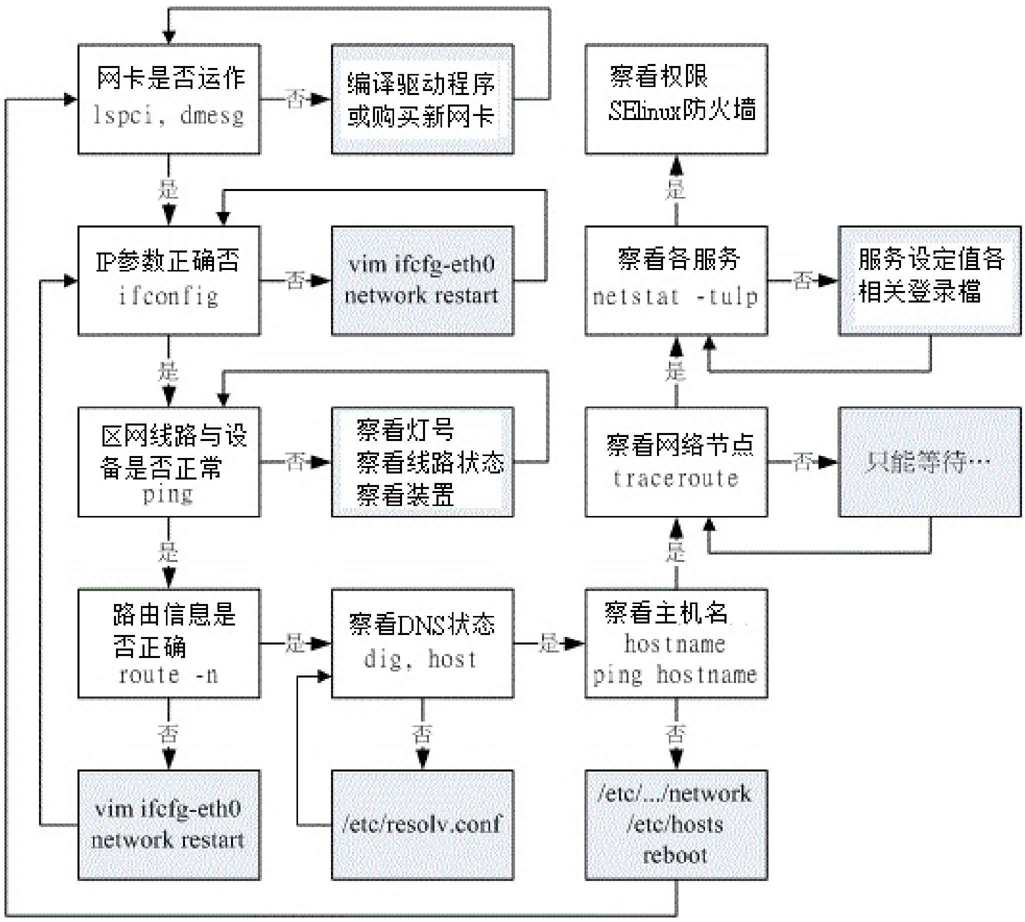


图 6.2-1、网络问题解决流程图

6.3 本章习题

- 以图 6.1-1 的星形联机为例，你的 Linux PC 3 可以 ping 到 Windows PC1 ，但是反过来，Windows PC1 无法 ping 到 Linux PC3，你觉得原因可能发生在哪里？由于两边已经可以用 ping 进行联机，所以硬件应该是没有问题了。而 Linux --> Windows 没问题，Windows --> Linux 有问题，可能是由于 Linux 主机上面的防火墙所致。可以使用 iptables -L -n 去查阅一下防火墙的设定规则。详细的防火墙请参考后续的章节。

6.4 参考数据与延伸阅读

- 注1：网中人的网络架构简介：http://www.study-area.org/network/network_archi.htm

---

2002/07/31：第一次完成日期！ 2003/08/19：重新修订一些数据，与前面的章节比较好配合！ 2006/08/04：将旧的文章移动到 此处 2010/09/03：将旧的基于 CentOS 4.x 所撰写的版本移动到此处 2010/09/06：花了一些时间，加上一张流程图分析！参考看看吧！
2011/07/19：将版本改为 CentOS 6.x，原本的 CentOS 5.x 放于 此处

第二部分：主机的简易资安防护措施

有很多团体做过许多操作系统安全性侦测的研究，他们发现一部没有经过更新与保护的 Linux/Windows 主机 (不论是一般个人计算机还是服务器)，只要一接上 Internet 几乎可以在数小时以内就被入侵或被当成跳板！ 您瞧瞧，这是啥世界啊～所以说，要好好的保护好您自己的服务器主机才行喔！那应该要如何保护你的服务器主机呢？ 基本上，你最要知道的是你的服务器开了多少网络服务，而这些服务会启动什么埠口？ 根据这层关系来关闭一些不必要的网络服务。再者，利用在线更新系统让你的 Linux 随时保持在最新的软件的状态， 这个小动作可以预防绝大部分的入侵攻击，可以说是最重要的一步了！最后才是架设基础防火墙。

因为 Linux 的功能太强了，如果你不好好的保护好你的主机，要是被入侵并且被当成跳板，这可能会让您吃上官司的！ 不要小看这层动作喔！虽然被入侵后只要将旧系统移除并且重灌后，你的服务器主机就能够『短暂』的恢复正常， 不过如果您的一些操作习惯不改的话，呵呵，并不是重灌就能够让你的服务器主机活的好好的喔！ 所以啰，我们在架站之前，基本的网络防备措施还是得来了解一下，免得经常要重灌、重灌、重灌....

第七章、网络安全与主机基本防护：限制端口, 网络升级与 SELinux

最近更新日期：2011/07/21

通过第一篇的锻炼之后，现在你应该已经利用 Linux 连上 Internet 了。但是你的 Linux 现在恐怕还是不怎么安全的。 因此，在开始服务器设定之前，我们必须要让你的系统强壮些！以避免被恶意的 cracker 所攻击啊！在这一章当中， 我们会介绍封包的流向，然后根据该流向来制订系统强化的流程！包括在线自动升级、服务管控以及 SELinux 等等。现在就来了解了解啰！

7.1 网络封包联机进入主机的流程

在这一章当中，我们要讨论的是，当来自一个网络上的联机要求想进入我们的主机时， 这个网络封包在进入主机实际取得数据的整个流程是如何？了解了整个流程之后， 你才会发现：原来系统操作的基本概念是如此的重要！ 而你也才会了解要如何保护你的主机安全呐！闲话少说，咱们赶紧来瞧一瞧先。

7.1.1 封包进入主机的流程

在第一章我们就谈过网络联机的流程， 当时举的例子是希望你可以理解为啥架设服务器需要了解操作系统的基本观念。在这一章当中，我们要将该流程更细致化说明， 因为，透过这个流程分析，你会知道为啥我们的主机需要进行过一些防护之后，系统才能够比较强壮。此外，透过第二章的网络概念解释后，你也了解了网络是双向的，服务器与客户端都得要有 IP:port 才能够让彼此的软件互相沟通。那么现在，假设你的主机是 WWW 服务器，透过底下的图标，网络封包如何进入你的主机呢？

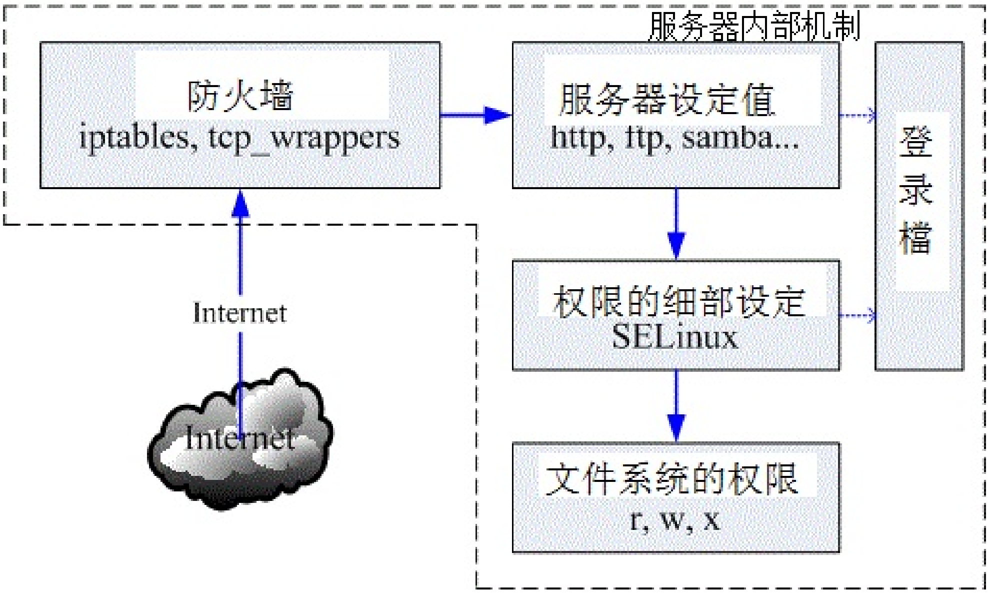


图 7.1-1、网络封包进入主机的流程

1. 经过防火墙的分析：

   Linux 系统有内建的防火墙机制，因此你的联机能不能成功，得要先看防火墙的脸色才行。预设的 Linux 防火墙就有两个机制，这两个机制都是独立存在的，因此我们预设就有两层防火墙喔。第一层是封包过滤式的 netfilter 防火墙， 另一个则是透过软件控管的 TCP Wrappers 防火墙。

- 封包过滤防火墙：IP Filtering 或 Net Filter 要进入 Linux 本机的封包都会先通过 Linux 核心的预设防火墙，就是称为 netfilter 的咚咚，简单的说，就是 iptables 这个软件所提供的防火墙功能。为何称为封包过滤呢？因为他主要是分析 TCP/IP 的封包表头来进行过滤的机制，主要分析的是 OSI 的第二、三、四层，主要控制的就是 MAC, IP, ICMP, TCP 与 UDP 的埠口与状态 (SYN, ACK...) 等。详细的资料我们会在第九章防火墙介绍。

- 第二层防火墙：TCP Wrappers 通过 netfilter 之后，网络封包会开始接受 Super daemons 及 TCP_Wrappers 的检验，那个是什么呢？ 说穿了就是 /etc/hosts.allow 与 /etc/hosts.deny 的配置文件功能啰。 这个功能也是针对 TCP 的 Header 进行再次的分析，同样你可以设定一些机制来抵制某些 IP 或 Port ，好让来源端的封包被丢弃或通过检验；

  透过防火墙的管控，我们可以将大部分来自因特网的垃圾联机丢弃，只允许自己开放的服务的联机进入本机而已， 可以达到最基础的安全防护。

2. 服务 (daemon) 的基本功能：

   预设的防火墙是 Linux 的内建功能，但防火墙主要管理的是 MAC, IP, Port 等封包表头方面的信息，如果想要控管某些目录可以进入， 某些目录则无法使用的功能，那就得要透过权限以及服务器软件提供的相关功能了。举例来说，你可以在 httpd.conf 这个配置文件之内规范某些 IP 来源不能使用 httpd 这个服务来取得主机的数据， 那么即使该 IP 通过前面两层的过滤，他依旧无法取得主机的资源喔！但要注意的是， 如果 httpd 这支程序本来就有问题的话，那么 client 端将可直接利用 httpd 软件的漏洞来入侵主机，而不需要取得主机内 root 的密码！因此， 要小心这些启动在因特网上面的软件喔！

3. SELinux 对网络服务的细部权限控制：

   为了避免前面一个步骤的权限误用，或者是程序有问题所造成的资安状况，因此 Security Enhanced Linux (安全强化 Linux) 就来发挥它的功能啦！简单的说，SELinux 可以针对网络服务的权限来设定一些规则 (policy) ，让程序能够进行的功能有限， 因此即使使用者的档案权限设定错误，以及程序有问题时，该程序能够进行的动作还是被限制的，即使该程序使用的是 root 的权限也一样。举例来说，前一个步骤的 httpd 真的被 cracker 攻击而让对方取得 root 的使用权，由于 httpd 已经被 SELinux 控制在 /var/www/html 里面，且能够进行的功能已经被规范住了，因此 cracker 就无法使用该程序来进行系统的进一步破坏啰。现在这个 SELinux 一定要开启喔！

4. 使用主机的文件系统资源：

   想一想，你使用浏览器连接到 WWW 主机最主要的目的是什么？当然就是读取主机的 WWW 数据啦！ 那 WWW 资料是啥？就是档案啊！^_^！所以，最终网络封包其实是要向主机要求文件系统的数据啦。 我们这里假设你要使用 httpd 这支程序来取得系统的档

案数据，但 httpd 默认是由一个系统账号名称为 httpd 来启动的，所以：你的网页数据的权限当然就是要让 httpd 这支程序可以读取才行啊！如果你前面三关的设定都 OK ，最终权限设定错误，使用者依旧无法浏览你的网页数据的。

在这些步骤之外，我们的 Linux 以及相关的软件都可能还会支持登录文件记录的功能，为了记录历史历程， 以方便管理者在未来的错误查询与入侵检测，良好的分析登录档的习惯是一定要建立的，尤其是 /var/log/messages 与 /var/log/secure 这些个档案！虽然各大主要 Linux distribution 大多有推出适合他们自己的登录文件分析软件，例如 CentOS 的 logwatch ，不过毕竟该软件并不见得适合所有的 distributions ，所以鸟哥尝试自己写了一个 logfile.sh 的 shell script，你可以在底下的网址下载该程序：

- http://linux.vbird.org/download/index.php?action=detail&fileid=60

好了，那么根据这些流程，你觉得 cracker 这些个坏蛋能够怎样的攻击我们的系统呢？得要先到对方想要怎么破坏， 我们才能够想办法来补强系统嘛！底下先讲讲基本的攻击手法啰。

7.1.2 常见的攻击手法与相关保护

我们由图 7.1-1 了解到数据传送到本机时所需要经过的几道防线后，那个权限是最后的关键啦！ 现在你应该比较清楚为何我们常常在基础篇里面一直谈到设定正确的权限可以保护你的主机了吧？ 那么 cracker 是如何透过上述的流程还能够攻击你的系统啊？底下就让我们来分析分析。

- 取得帐户信息后猜密码

由于很多人喜欢用自己的名字来作为帐户信息，因此账号的取得是很容易的！举例来说，如果你的朋友将你的 email address 不小心泄漏出去，例如： dmtsai@your.host.name 之类的样式，那么人家就会知道你有一部主机，名称为 your.host.name，且在这部主机上面会有一个使用者账号，账号名称为 dmtsai ，之后这个坏家伙再利用某些特殊软件例如 nmap 来进行你主机的 port scan 之后，嘿嘿！他就可以开始透过你主机有启动的软件功能来猜你这个账号的密码了！

另外，如果你常常观察你的主机登录文件，那你也会发现如果你的主机有启动 Mail server 的服务时， 你的登录档就会常常出现有些怪家伙尝试以一些奇怪的常见账号在试图猜测你的密码， 举例来说像：admin, administrator, webmaster .... 之类的账号，尝试来窃取你的私人信件。 如果你的主机真的有这类的账号，而且这类的账号还没有良好的密码规划，那就容易『中标』！ 唉！真是麻烦！所以我们常讲，系统账号千万不能给予密码，容易被猜密码啊！

这种猜密码的攻击方式算是最早期的入侵模式之一了，攻击者知道你的账号，或者是可以猜出来你的系统有哪些账号， 欠缺的就只是密码而已，因此他会『很努力的』去猜你的密码，此时，你的密码规划如果不好的话，很容易就被攻击了！ 主机也很容易被绑架啊！所以，良好的密码设置习惯是很重要的。

不过这种攻击方式比较费时，因为目前很多软件都有密码输入次数的限制，如果连续输入三次密码还不能成功的登入， 那该次联机就会被断线！所以，这种攻击方式日益减少，目前偶而还会看到就是了！这也是初级 cracker 会使用的方式之一。 那我们要如何保护呢？基本方式是这样的：

- 减少信息的曝光机会：例如不要将 Email Address 随意散布到 Internet 上头；
- 建立较严格的密码设定规则：包括 /etc/shadow, /etc/login.defs 等档案的设定， 建议你可以参考基础篇内的账号管理那一章来规范你的用户密码变更时间等等， 如果主机够稳定且不会持续加入某些账号时，也可以考虑使用 chattr 来限制账号 (/etc/passwd, /etc/shadow) 的更改；
- 完善的权限设定：由于这类的攻击方式会取得你的某个使用者账号的登入权限， 所以如果你的系统权限设定得宜的话，那么攻击者也仅能取得一般使用者的权限而已，对于主机的伤害比较有限啦！ 所以说，权限设定是重要的；

- 利用系统的程序漏洞『主动』攻击

由图 7.1-1 里面的第二个步骤中，我们知道如果你的主机有开放网络服务时， 就必须有启动某个网络软件嘛！我们也知道由于软件可能撰写方式的问题，可能产生一些会被 cracker 乱用的臭虫程序代码，而这些臭虫程序代码由于产生问题的大小，有分为 bug (臭虫，可能会造成系统的不稳定或当机) 与 Security (安全问题，程序代码撰写方式会导致系统的权限被恶意者所掌握) 等问题。

当程序的问题被公布后，某些较高阶的 cracker 会尝试撰写一些针对这个漏洞的攻击程序代码， 并且将这个程序代码放置到 cracker 常去的网站上面，藉以推销自己的『功力』..... 鸟哥要提醒的是，这种程序代码『是很容易被取得的』。 当更多『盈盈美黛子(台语，闲闲没事干之意)』取得这些程序代码后，他可能会想要『试一试这个攻击程序的威力』， 所以就拿来『扫射』一番，如果你八字比较轻，或者当天星座学家说你比较倒霉时，可能就会被不小心的攻击到...

这种攻击模式是目前最常见的，因为攻击者只要拿到攻击程序就可以进行攻击了， 『而且由攻击开始到取得你系统的 root 权限不需要猜密码， 不需要两分钟，就能够立刻入侵成功』， 所以『盈盈美黛子』们最爱的就是这个咚咚了。 但这个玩意儿本身是靠『你主机的程序漏洞』来攻击的，所以，如果你的主机随时保持在实时更新的阶段， 或者是关闭大部分不需要的程序，那就可以躲避过这个问题。因此，你应该要这样做：

- 关闭不需要的网络服务：开的 port 越少，可以被入侵的管道越少， 一部主机负责的服务越单纯，越容易找出问题点。
- 随时保持更新：这个没话讲！一定要进行的！
- 关闭不需要的软件功能：举例来说，后面会提到的远程登录服务器 SSH 可以提供 root 由远程登录，那么危险的事情当然要给他取消啊！^_^

- 利用社交工程作欺骗

社交工程 (Social Engineering) 指的其实很简单，就是透过人与人的互动来达到『入侵』的目的！ @_@！人与人的互动可以入侵你的主机？鸟哥在呼咙你吗？当然不是。

近日在台湾的社会你不是常看到某些人会以『退税、中奖、花小钱买贵重物品』等名义来欺骗善良老百姓， 让老百姓掏出口袋里的金钱给那些可恶的金光党吗？社交工程也是类似的方法。在大公司里面， 或许你可能会接到这样的电话：『我是人事部门的经理，我的账号为何突然间不能登入了？ 你给我看一看，恩？干脆直接帮我另建一个账号，我告诉你我要的密码是....』。如果你一时不查给他账号密码的话， 你的主机可能就这样被绑走了～

社交工程的欺骗方法多的是，包括使用『好心的 email 通知』、『警告信函』、『中奖单』等等， 在在都是要欺骗你的账号密码，有的则利用钓鱼方式来欺骗你在某些恶意网站上面输入你的账号密码， 很讨厌的啦！举例来说，我们昆山计中的 email 常常会收到系统维护的信件，要我们将账号密码提交给系统管理员统一控管， 这当然是假的！计中根本不会寄出这样的信件啊！伤脑筋啦！所以要注意啊！那要如何防范呢？

- 追踪对谈者：不要一味的相信对方，你必须要有信心的向上呈报， 不要一时心慌就中了计！
- 不要随意透露账号/密码等信息：最好不要随意在 Internet 上面填写这些数据， 真的很危险的！因为在 Internet 上面，你永远不知道对方屏幕前面坐着的是谁？

- 利用程序功能的『被动』攻击

啥？除了主动攻击之外，还有所谓的被动攻击喔？没错啊，『系金乀』！那如何作被动攻击呢？ 那就得要由『恶意网站』讲起了。如果你喜欢上网随意浏览的话，那么有的时候可能会连上一些广告很多， 或者是一堆弹出式窗口的网站，这些网站有时还会很好心的『提供你很多好用的软件自动下载与安装』的功能， 如果该网站是你所信任的，例如 Red Hat, CentOS, Windows 官网的话，那还好， 如果是一个你也不清楚他是干嘛的网站，那你是否要同意下载安装该软件？

如果你常常在注意一些网络危机处理的相关新闻时，常会发现 Windows 的浏览器 (IE) 有问题， 有时则是全部的浏览器 (Firefox, Netscap, IE...) 都会出现问题。那你会不会觉得奇怪啊， 怎么『浏览器也会有问题？』这是因为很多浏览器会主动的答应对方 WWW 主机所提供的各项程序功能， 或者是自动安装来自对方主机的软件，有时浏览器还可能由于程序发生安全问题， 让对方 WWW 浏览器得以传送恶意代码给你的主机来执行，嘿嘿！中标！

那你又会想啊，那我干嘛浏览那样的恶意网站？呵呵！总是会有些粗心大意的时候啊！如果你今天不小心收到一个 email ，里面告诉你你的银行账号有问题，希望你赶紧连上某个网页去看看你的账号是否在有问题的行列中，你会不会去？ 如果今天有个网络消息说某某网页在提供大特价商品，那你会不会去碰碰运气？ 都是可能的啊！不过，这也就很容易被对方攻击到了。

那如何防备啊？当然建立良好的习惯最重要了：

- 随时更新主机上的所有软件：如果你的浏览器是没有问题的， 那对方传递恶意代码时，

你的浏览器就不会执行，那自然安全的多啊！
- 较小化软件的功能：举例来说，让你的收信软件不要主动的下载文件， 让你的浏览器在安装某些软件时，要通过你的确认后才安装，这样就比较容易克服一些小麻烦；
- 不要连接到不明的主机：其实鸟哥认为这个才最难！ 因为很多时候我们都用 google 在搜寻问题的解决之道啊，那你如何知道对方是否是骗人的？ 所以，前面两点防备还是很重要的！不要以为没有连接上恶意网站就不会有问题啊！

- 蠕虫或木马的 rootkit

rootkit 意思是说可以取得 root 权限的一群工具组 (kit)，就如同前面主动攻击程序漏洞的方法一样， rootkit 主要也是透过主机的程序漏洞。不过， rootkit 也会透过社交工程让用户下载、安装 rootkit 软件， 结果让 cracker 得以简单的绑架对方主机啊！

rootkit 除了可以透过上述的方法来进行入侵之外，rootkit 还会伪装或者是进行自我复制， 举例来说，很多的 rootkit 本身就是蠕虫或者是木马间谍程序。蠕虫会让你的主机一直发送封包向外攻击， 结果会让你的网络带宽被吃光光，例如 2001-2003 年间的 Nimda, Code Red 等等；至于木马程序 (Trojan Horse) 则会对你的主机进行开启后门 (开一个 port 来让 cracker 主动的入侵)，结果就是....绑架、绑架、绑架！

rootkit 其实挺不好追踪的，因为很多时候他会主动的去修改系统观察的指令， 包括 ls, top, netstat, ps, who, w, last, find 等等，让你看不到某些有问题的程序， 如此一来，你的 Linux 主机就很容易被当成是跳板了！有够危险！那如何防备呢？

- 不要随意安装不明来源的档案或者是不明网站的档案数据；
- 不要让系统有太多危险的指令：例如 SUID/SGID 的程序， 这些程序很可能会造成用户不当的使用，而使得木马程序有机可趁！
- 可以定时以 rkhunter 之类的软件来追查：有个网站提供 rootkit 程序的检查，你可以前往下载与分析你的主机： http://www.rootkit.nl/projects/rootkit_hunter.html

- DDoS 攻击法 (Distributed Denial of Service)

这类型的攻击中文翻译成『分布式阻断服务攻击』，从字面上的意义来看，它就是透过分散在各地的僵尸计算机进行攻击， 让你的系统所提供的服务被阻断而无法顺利的提供服务给其他用户的方式。 这种攻击法也很要命，而且方法有很多，最常见的就属 SYN Flood 攻击法了！还记得我们在网络基础里面提到的，当主机接收了一个带有 SYN 的 TCP 封包之后，就会启用对方要求的 port 来等待联机，并且发送出回应封包 (带有 SYN/ACK 旗目标 TCP 封包)，并等待 Client 端的再次回应。

好了，在这个步骤当中我们来想一想，如果 cient 端在发送出 SYN 的封包后，却将来自 Server 端的确认封包丢弃，那么你的 Server 端就会一直空等，而且 Client 端可以透过软件功能，在短短的时间内持续发送出这样的 SYN 封包，那么你的 Server 就会持续不断的发送确认封包，并且开启大量的 port 在空等～呵呵！等到全部主机的 port 都启用完毕，那么.....系统就挂了！

更可怕的是，通常攻击主机的一方不会只有一部！他会透过 Internet 上面的僵尸网络 (已经成为跳板，但网站主却没有发现的主机) 发动全体攻击，让你的主机在短时间内就立刻挂点。 这种 DDoS 的攻击手法比较类似『玉石俱焚』的手段， 他不是入侵你的系统，而是要让你的系统无法正常提供服务！ 最常被用来作为阻断式服务的网络服务就是 WWW 了，因为 WWW 通常得对整个 Internet 开放服务。

这种攻击方法也是最难处理的，因为要嘛就得要系统核心有支持自动抵挡 DDoS 攻击的机制， 要嘛你就得要自行撰写侦测软件来判断！真是麻烦啊～而除非你的网站非常大， 并且『得罪不少人』，否则应该不会被 DDoS 攻击啦！ ^_^

- 其他

上面提到的都是比较常见的攻击方法，是还有一些高竿的攻击法啦，不过那些攻击法都需要有比较高的技术水准，例如 IP 欺骗。他可以欺骗你主机告知该封包来源是来自信任网域，而且透过封包传送的机制， 由攻击的一方持续的主动发送出确认封包与工作指令。如此一来，你的主机可能就会误判该封包确实有响应， 而且是来自内部的主机。

不过我们知道因特网是有路由的，而每部主机在每一个时段的 ACK 确认码都不相同， 所以这个方式要达成可以登入，会比较麻烦，所以说，不太容易发生在我们这些小型主机上面啦！ 不过你还是得要注意一下说：

- 设定规则完善的防火墙：利用 Linux 内建的防火墙软件 iptables 建立较为完善的防火墙，可以防范部分的攻击行为；
- 核心功能：这部份比较复杂，你必须要对系统核心有很深入的了解， 才有办法设定好你的核心网络功能。
- 登录文件与系统监控：你可以透过分析登录文件来了解系统的状况， 另外也可以透过类似 MRTG 之类的监控软件 来实时了解到系统是否有异常，这些工作都是很好的努力方向！

- 小结语

要让你的系统更安全，没有『三两三』是没办法『上梁山』的！我们也一直鼓吹， 『维护网站比架设网站还要重要』的观念！因为『一人得道鸡犬升天』，同样的道理：『一人中标全员挂点』， 不要以为你的主机没有啥重要数据，被入侵或被植入木马也没有关系， 因为我们的服务器通常会对内部来源的主机规范的较为宽松，如果你的主机在公司内部， 但是不小心被入侵的话，那么贵公司的服务器是否就会暴露在危险的环境当中了？

另外，在蠕虫很『发达』的年代，我们也会发现只要局域网络里面有一部主机中标，整个局域网络就会无法使用网络了， 因为带宽已经被蠕虫塞爆！如果老板发现他今天没有办法收信了，但无法收信的原因并非服务器挂点， 而是因为内部人员的某部个人计算机中了蠕虫，而那部主机中蠕虫的原因只是因为该使用者不小心去看了一下色情网站， 你觉得老板会高兴的跟该员工一起看色情网站还是 fire 掉该人员？

所以啊，主机防护还是很重要的！不要小看了！提供几个方向给大家思考看看吧：

1. 建立完善的登入密码规则限制；
2. 完善的主机权限设定；
3. 设定自动升级与修补软件漏洞、及移除危险软件；
4. 在每项系统服务的设定当中，强化安全设定的项目；
5. 利用 iptables, TCP_Wrappers 强化网络防火墙；
6. 利用主机监控软件如 MRTG 与 logwatch 来分析主机状况与登录文件；

7.1.3 主机能作的保护：软件更新、减少网络服务、启动 SELinux

根据本章前面的分析，现在你知道封包的流向以及主机基本需要进行的防护了。不过你或许还是有疑虑，那就是， 既然我都已经有了防火墙，那么权限的控管啦、密码的严密性啦、服务器软件的更新啦、SELinux 啦等等的， 是否就没有这么重要呢？毕竟它是封包进入的第一关卡！这关把关严格，后续可以稍微宽松吗？其实...你错了！ 对于开放某些服务的服务器来说，你的防火墙『根本跟屁一样，是没有用的！』怎么说呢？

- 软件更新的重要性

让我们瞧一瞧图 7.1-1 的流程好了，假设你需要对全世界开放 WWW ，那么提供 WWW 服务的 httpd 这只程序就得要执行，并且，你的防火墙得要打开 port 80 让全世界都可以连接到你的 port 80 ，这样才是一部合理的 WWW 服务器嘛！问题来啦，如果 httpd 这只程序有资安方面的问题时，请问防火墙有没有效用？当然没有！因为防火墙原本就得要开放 port 80 啊！此时防火墙对你的 WWW 一点防护也没有。那怎办？

没啥好说的，就是软件持续更新到最新就对了！因为自由软件就是有这个好处，当你的程序有问题时， 开发商会在最短的时间内取得志工提供的修补程序 (patch) ，并将该程序代码补充到软件更新数据库中， 让一般用户可以直接透过网络来自动更新。因此，要克服这个服务器软件的问题，更新系统软件就对了。

但是你得要注意，你的系统能否更新软件与系统的版本有关！举例来说，2003 年左右发布的 Red Hat 9 目前已经没有支持了， 如果你还是执意要安装 Red Hat 9 这套系统，那么很抱歉，你得要手动将系统内的软件透过 make 动作来重新编译到最新版， 因此，很麻烦～同样的， Fedora 最新版虽然有提供网络自动更新，但是 Fedora 每一个版本的维护期间较短， 你可能需要常常大幅度的变更你的版本，这对服务器的设定也不妥当。此时一个企业版本的 Linux distributions 就很重要啦！ 举例来说，鸟站的主机截至目前为止 (2011/07) 还是使用 CentOS 4.x ，因为这个版本目前还是持续维护中。 这对服务器来说，是相当重要的！稳定与安全比什么都重要！

想要了解软件的安全通报，可以参考如下的网站数据喔！

- 台湾计算机危机处理小组(TWCERT)：http://www.cert.org.tw/
- Red Hat 的官方说明：https://www.redhat.com/support/

- 认识系统服务的重要性

再回到图 7.1-1 当中，同时思考一下第二章网络基础里面谈到的网络联机是双向这件事， 我们会得到一个答案，那就是在图 7.1-1 内的第二个步骤中，如果能够减少服务器上面的监听埠口， 此时因为服务器端没有可供联机的埠口，客户端当然也就无法联机到服务器端嘛！那么如何限制服务器开启的埠口呢？ 第二章就谈到过了，关闭埠口的方式是透过关闭网络服务。没错啊！所以啰，此时能够减少网络服务就减少，可以避免很多不必要的麻烦。

- 权限与 SELinux 的辅助

根据网络上面多年来的观察，很多朋友在发生权限不足方面的问题后，都会直接将某个目录直接修订成为 chmod -R 777 /some/path/。 如果这部主机只是测试用的没有上网提供服务，那还好。如果有上网提供某些服务时，那可就伤脑筋了！因为目录的 wx 权限设定一起后，代表该身份可以进行新增与删除的动作。偏偏你又给 777 (rwxrwxrwx) ，代表所有的人都可以在该目录下进行新增与删除！ 万一不小心某支程序被攻击而被取得操作权，想想看，你的系统不就可能被写入某些可怕的东西了吗？ 所以不要随便设定权限啊！

那如果由于当初规划的账号身份与群组设定的太杂乱，导致无法使用单纯的三种身份的三种权限来设定你的系统时，那该如何是好？ 没关系的，可以透过 ACL 这个好用的东西！ ACL 可以针对单一账号或单一群组进行特定的权限设定，相当好用喔！ 他可以辅助传统 Unix 的权限设定方面的困扰哩。详情请参考基础篇的内容呦！

那如何避免用户乱用系统，乱设定权限呢？这个时候就得要透过 SELinux 来控制了。 SELinux 可以在程序与档案之间再加入一道细部的权限控制，因此，即使程序与档案的权限符合了操作动作，但如果程序与档案的 SELinux 类型 (type) 不吻合时，那么该程序就无法读取该档案喔！ 此外，我们的 CentOS 也针对了某些常用的网络服务制订了许多的档案使用规则 (rule)，如果这些规则没有启用， 那么即使权限、SELinux 类型都对了，该网络服务的功能还是无法顺利的运作喔！

根据这样的分析，我们可以知道，随时更新系统软件、限制联机端口以及透过启动 SELinux 来限制网络服务的权限，经过这三个简单的步骤，你的系统将可以获得相当大的保护！当然啦， 后续的防火墙以及系统注册表档分析工作仍是需要进行的。本章后续将依据这三点来深入介绍。

7.2 网络自动升级软件

在现在的因特网上面，cracker 实在是太多了！这些闲人会利用已经存在的系统漏洞，来进行侦测、入侵你的主机。 因此，除了未来架设防火墙之外，最重要的 Linux 日常管理工作，莫过于软件的升级了！ 不过，如果使用者还得要自己每天观察网络安全通报，并主动去查询各大 distribution 针对这些漏洞来提供升级软件包， 那真是太不人性化了！因此，目前就有很多在线直接更新的机制出现了！有了这些在线直接更新软件的手段与方法， 我们系统管理员在管理主机系统上面，可就轻松的多啰！

7.2.1 如何进行软件升级

通常鸟哥安装好 Linux 之后，会先开启系统默认的防火墙机制，然后第一件事情就是进行全系统更新啦！ 不论是哪一套 Linux 鸟哥都是这样做的，因为要避免软件资安的问题嘛！好了，那么 Linux 上面的软件该如何进行更新与升级呢？ 还记得你是如何安装软件的吗？不就是 rpm, tarball 与 dpkg 吗？ 所以啰，你的软件如果想要升级，那就得依据当时你安装该软件的方式来进行升级啊！而每种方式都有其适用性：

- RPM： 这是目前最常见于 Linux distribution 当中的软件管理方式，包括 CentOS / Fedora / SuSE / Red Hat / Mandriva 等等，都是使用这个方式来管理的；
- Tarball： 利用软件的官方网站所释出的原始码在您的系统上面编译与安装， 一般来说，由于软件是直接在自己的机器上面编译的，所以效能会比较好一些。 不过，升级的时候就比较麻烦，因为又得要下载新的原始码并且重新编译一次。 这种安装模式常见于某些特殊软件 (没有包含在 distribution 当中)，或者是 Gentoo 这个强调效能的 distribution；
- dpkg： 是 debian 这个 distribution 所使用的软件管理方式，与 RPM 很类似，都是透过预先编译的处理，可以让 end user 直接使用来升级与安装。

举例来说，如果你的系统是 CentOS ，我们知道他使用的是 RPM 类型的软件管理模式，那如果你想要安装 B2D 的软件怎办？要注意， B2D 是使用 debian 的 dpkg 来管理软件的，两者并不相同啊！要互相安装太难了！ 所以说，要升级的话，得先了解到你系统上的软件安装与管理的方法才行。

不过，有个特殊案例，那就是旧版本的 Linux (例如 Red Hat 9) 的软件升级该如何是好？ 由于旧版本的软件支持度本来就比较差，商业公司或者是社群也没有这么多心力放在旧版本的支持上， 所以，你这个时候可以选择： (1)升级到较新的版本，例如 CentOS 6.x，或者是 (2) 利用 Tarball 来自行升级核心与软件。不过，比较建议升级到新版本啦，因为要自行以手动方式由 Tarball 安装到最新的版本，实在是很费时费力，而且还得要常常查阅官方网站所推出的最新消息， 漏过一则都可能发生无法预期的状况。

我们都晓得在 Windows 的环境下，他有提供一个 Live update 的项目可以自动的在线升级，甚至很多的防病毒软件与防木马软件也都有推出实时的在线更新，如此一来可以让您的软件维持在最新版的状况， 眞是好啊！咦！那我们的 Linux 是否有这样的功能？如果有的话，那么系统自动进行软件升级， 不就可以轻松又快乐了？没错！确实是这样的！所以就让我们来谈一谈 Linux 的在线升级机制吧！

在 Linux 最常见的软件安装方式： RPM / Tarball / dpkg 当中，Tarball 由于取得的是原始码，所以要用 Tarball 来作在线自动更新是不太可能进行的，所以仅能用 RPM 或 dpkg 这两种软件管理的方式来进行在线更新了。

但 RPM 与 dpkg 不是有所谓的相依属性吗？这倒不需要担心呐！因为我们的 RPM 与 dpkg 软件档案都有一些软件的基本信息， 并同时记录了软件的相依属性 (记得使用 rpm -q 的查询吗)，所以当分析这些基本信息并使用一些机制将这些相依信息记录下来后， 再透过一些额外的网络功能，就能够自动的分析你的系统与修补软件之间的差异， 并可进一步帮你分析所需要升级与相依属性的软件，就可达成自动升级的理想啦！

由于各家 distributions 在管理系统上都有自己独特的想法，所以在分析 RPM 或 dpkg 软件与方式上面就有所不同， 也就有底下这些不同的在线升级机制啦：

- yum： CentOS 与 Fedora 所常用的自动升级机制，透过 FTP 或 WWW 来进行在线升级以及在线直接安装软件；
- apt： 最早由 debian 这个 distribution 所发展，现在 B2D 也是使用 apt ，同时由于 apt 的可移植性， 所以只要你的 RPM 可以使用 apt 来管理的话，就可以自行建立 apt 服务器来提供其他使用者进行在线安装与升级。
- you： 所谓的 Yast Online Update (YOU) 是由 SuSE 所自行开发出来的在线安装升级方式， 经过注册取得一组账号密码后，就能够使用 you 的机制来进行在线升级。不过如果是免费的版本， 则仅有 60 天的试用期！
- urpmi： 这个则是 Mandriva 所提供的在线升级机制！

讲了这些升级机制并且与 distribution 作了对应，你就该了解到：『每个 distribution 可以使用的在线升级机制都不相同』的啊！所以请参考你的 distribution 所提供的文件来进行在线升级的设定喔！否则就得要自行手动下载安装了！ @_@

鸟哥这里都是使用 CentOS 这个 Red Hat 兼容的 distributions 来介绍的，因此，底下仅介绍了 yum 而已。 不过，yum 已经能够适用于 CentOS, Red Hat Enterprise Linux, Fedora 等等，也应该是挺够用的了！ 另外，基础篇里面已经谈过 rpm 与 yum 的用法，所以在这里仅是加强介绍与更新有关的用法而已喔！

7.2.2 CentOS 的 yum 软件更新、映像站使用的原理

我们曾经在基础篇里面谈过 yum 了，基本上他的原理是，我们的 CentOS 会跑到 yum 服务器上头，下载了官方网站释出的 RPM 表头列表数据，该数据除了记载每个 RPM 软件的相依性之外，也说明了 RPM 档案所放置的容器 (repository) 所在。因此透过分析这些数据，我们的 CentOS 就能够直接使用 yum 去下载与安装所需要的软件了！ 详细图标与流程有点像这样：

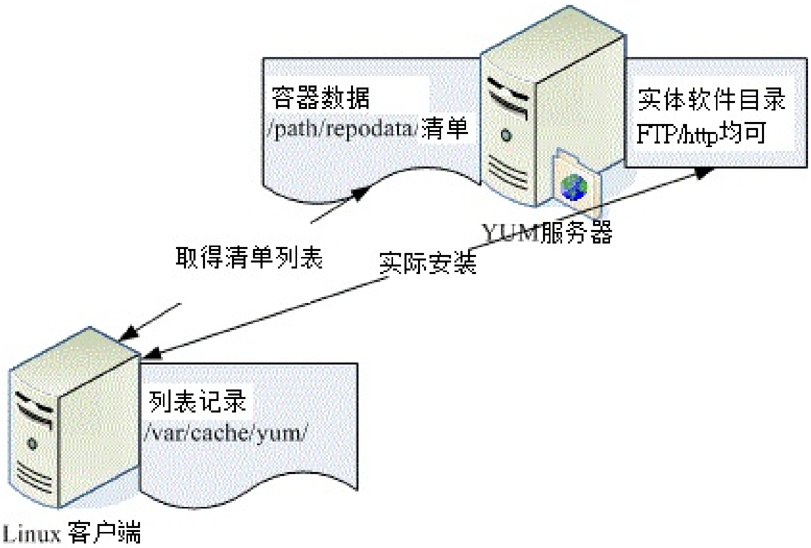


图 7.2-1、使用 yum 下载清单表头与取得容器相关资料示意图

1. 先由配置文件判断 yum server 所在 IP 地址；
2. 连接到 yum server 后，先下载新的 RPM 档案的表头数据；
3. 分析比较使用者所欲安装/升级的档案，并提供使用者确认；
4. 下载用户选择的档案到系统中的 /var/cache/yum ，并进行实际安装；

由于你所下载的清单当中已经含有所有官方网站所释出的 RPM 档案的表头相依属性的关系，所以如果你想要安装的软件包含某些尚未安装的相依软件时，我们的 yum 会顺便帮你下载所需要的其他软件，预安装后， 再安装你所实际需要的软件！从分析、下载到安装，全部一口气搞定！很简单的啦！

不过，恐怕还是有问题。如果全世界使用 CentOS 的朋友通通联机到同一部 Yum 服务器去下载所需要的 RPM 档案，哇！ 那带宽不就很容易被塞爆！那怎办？没关系，有所谓的映射站啊！ CentOS 在世界各地都有映射站，这些映射站会将官网的 yum 服务器的数据复制一份，同时在映射站上面也提供同样的 yum 功能，因此，你可以在任何一部 yum 服务器的映像站上面下载与安装软件。底下是 CentOS 官网上面列出的亚洲地区映射站一览表：

- http://www.centos.org/modules/tinycontent/index.php?id=32

现在的 yum 又很聪明，它会自动的去分析离你的主机最近的那部映射站，然后直接使用该部映像主机作为你的 yum 来源， 因此，『理论上』你不需要更动任何设定，在台湾，你的 CentOS 就会使用台湾地区的 yum 服务器啰！就这么简单！ 所以，接下来就让我们直接来谈谈怎么使用 yum 吧！

Tips: yum 的原理与相关使用，我们在基础篇里面已经分门别类的介绍过了，因此底下仅就比较重要的部分介绍一下啰！



7.2.3 yum 的使用： 安装, 软件群组, 全系统更新

yum 可不止能够在线自动升级而已，他还可以作查询、软件群组的安装、整体版本的升级等等，好用的哩！ 先来谈论一下 yum 这个指令的用法吧：

```
[root@www ~]# yum [option] [查询的工作项目] [相关参数]
选项与参数：
option：主要的参数，包括有：
   -y ：当 yum 询问使用者的意见时，主动回答 yes 而不需要由键盘输入；

[查询的工作项目]：由于不同的使用条件，而有一些选择的项目，包括：
   install ：指定安装的软件名称，所以后面需接『 软件名称 』
   update  ：进行整体升级的行为；当然也可以接某个软件，仅升级一个软件；
   remove  ：移除某个软件，后面需接软件名称；
   search  ：搜寻某个软件或者是重要关键字；
   list    ：列出目前 yum 所管理的所有的软件名称与版本，有点类似 rpm -qa；
   info    ：同上，不过有点类似 rpm -qai 的执行结果；
   clean   ：下载的档案被放到 /var/cache/yum ，可使用 clean 将他移除，
             可清除的项目：packages &#124; headers &#124; metadata &#124; cache 等；

在[查询的工作项目]部分还可以具有整个群组软件的安装方式，如下所示：
   grouplist    ：列出所有可使用的『软件群组』，例如 Development Tools 之类的；
   groupinfo    ：后面接 group_name，则可了解该 group 内含的所有软件名；
   groupinstall：这个好用！可以安装一整组的软件群组，相当的不错用！
                 更常与 --installroot=/some/path 共享来安装新系统
   groupremove ：移除某个软件群组；

# 范例一：搜寻 CentOS 官网提供的软件名称是否有与 RAID 有关的？
[root@www ~]# yum search raid
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile  &lt;==这里就是在测试最快的映射站
 * base: ftp.isu.edu.tw                     &lt;==共有四个容器内容
 * extras: ftp.isu.edu.tw                   &lt;==每个容器都在 ftp.isu.edu.tw 上
 * updates: ftp.isu.edu.tw
base                           &#124; 3.7 kB     00:00  &lt;==下载软件的表头列表中
extras                         &#124;  951 B     00:00
updates                        &#124; 3.5 kB     00:00
=================== Matched: raid =================&lt;==找到的结果如下
dmraid.i686 : dmraid (Device-mapper RAID tool and library)
....(中间省略)....
mdadm.x86_64 : The mdadm program controls Linux md devices (software RAID
....(底下省略)....

# 范例二：上述输出结果中， mdadm 的功能为何？
[root@www ~]# yum info mdadm
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: ftp.twaren.net
 * extras: ftp.twaren.net
 * updates: ftp.twaren.net
Installed Packages  &lt;==这里说明这是已经安装的软件！
Name        : mdadm
Arch        : x86_64
Version     : 3.1.3
Release     : 1.el6
Size        : 667 k
Repo        : installed
From repo   : anaconda-CentOS-201106060106.x86_64
Summary     : The mdadm program controls Linux md devices (software RAID
URL         : http://www.kernel.org/pub/linux/utils/raid/mdadm/
License     : GPLv2+
Description: The mdadm program is used to create, manage, and monitor
....(底下省略)....
# 由上述底线的 Summary 关键词，知道这软件在达成软件磁盘阵列功能！！
```

yum 真是个很好用的东西，它可以直接查询是否有某些特殊的软件名称。举例来说，你可以利用底下的两个方式取得软件名称：

- yum search "一些关键词"
- yum list (可列出所有的软件文件名)

然后再以正规表示法取得关键词，或者是『 yum info "软件名称" 』就能够知道该软件的用途，最后再决定要不要安装啊！上面的范例一就是在找出磁盘阵列的管理软件。 如果确定要安装时，那就参考参考底下的流程吧！

- 利用 yum 进行安装

```
# 范例三：安装某个软件吧！以 mdadm 这个软件名为例：
[root@www ~]# yum install mdadm
....(前面省略)....
Setting up Install Process
Package mdadm-3.1.3-1.el6.x86_64 already installed and latest version
Nothing to do

[root@www ~]# yum install mdadma
Setting up Install Process
No package &lt;u&gt;mdadma available.&lt;/u&gt;
Nothing to do
```

仔细的看上述的两个指令，第二个指令鸟哥故意写错字，让软件名称由 mdadm 变成 mdadma 了！仿真同学如果打错字时所输出的讯息。由上述的讯息你可以知道，同样结果是『Nothing to do』，但是 yum 会告诉你该软件是『已安装 (installed and lastest version)』还是『没有该软件 (No package mdadma avaliable)』。作这个范例是希望朋友们能够仔细的看输出的讯息啦！好啦！我们还是来安装一个不曾装过的， 就拿 javacc 这套软件来装看看好了！

```
[root@www ~]# yum list javacc*
Available Packages
javacc.x86_64            4.1-0.5.el6      base
javacc-demo.x86_64       4.1-0.5.el6      base
javacc-manual.x86_64     4.1-0.5.el6      base
# 共有三套软件，分别是 javacc, javacc-demo, javacc-manual ，版本为 4.1-0.5.el6，
# 软件是放置到名称为 base 的容器当中存放的。

[root@www ~]# yum install javacc
....(前面省略)....
Setting up Install Process
Resolving Dependencies
--&gt; Running transaction check  &lt;==开始检查有没有相依属性的软件问题
---&gt; Package javacc.x86_64 0:4.1-0.5.el6 set to be updated
....(中间省略)....

=========================================================================
 Package                     Arch     Version           Repository  Size
=========================================================================
Installing:
 javacc                      x86_64   4.1-0.5.el6       base       895 k
Installing for dependencies:
 java-1.5.0-gcj              x86_64   1.5.0.0-29.1.el6  base       139 k
 java_cup                    x86_64   1:0.10k-5.el6     base       197 k
 sinjdoc                     x86_64   0.5-9.1.el6       base       705 k

Transaction Summary
=========================================================================
Install       4 Package(s)  &lt;==安装软件汇整，共安装 4 个，升级 0 个软件
Upgrade       0 Package(s)

Total download size: 1.9 M
Installed size: 5.6 M
Is this ok [y/N]: y  &lt;==让你确认要下载否！
Downloading Packages:
(1/4): java-1.5.0-gcj-1.5.0.0-29.1.el6.x86_64.rpm      &#124; 139 kB     00:00
(2/4): java_cup-0.10k-5.el6.x86_64.rpm                 &#124; 197 kB     00:00
(3/4): javacc-4.1-0.5.el6.x86_64.rpm                   &#124; 895 kB     00:00
(4/4): sinjdoc-0.5-9.1.el6.x86_64.rpm                  &#124; 705 kB     00:00
-------------------------------------------------------------------------
Total                                        3.1 MB/s &#124; 1.9 MB     00:00
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing     : java-1.5.0-gcj-1.5.0.0-29.1.el6.x86_64             1/4
  Installing     : 1:java_cup-0.10k-5.el6.x86_64                      2/4
  Installing     : sinjdoc-0.5-9.1.el6.x86_64                         3/4
  Installing     : javacc-4.1-0.5.el6.x86_64                          4/4

Installed:            &lt;==主要需要安装的
  javacc.x86_64 0:4.1-0.5.el6

Dependency Installed: &lt;==为解决相依性额外装的
  java-1.5.0-gcj.x86_64 0:1.5.0.0-29.1.el6   java_cup.x86_64 1:0.10k-5.el6
  sinjdoc.x86_64 0:0.5-9.1.el6

Complete!
```

瞧！经过 yum 我们可以很轻松的就安装好一个软件，并且这个软件已经主动的帮我们做好相依属性的克服了， 真是方便到爆！另外，CentOS 6.x 默认的情况下，yum 下载的数据除了每个容器的表头清单档案之外，所有下载的 RPM 档案都会在安装完毕之后予以删除！ 这样你的系统就不会有容量被下载的数据塞爆的问题。但如果你想要下载的 RPM 档案继续保留在 /var/cache/yum 当中，就得要修改 /etc/yum.conf 配置文件了！

```
[root@www ~]# vim /etc/yum.conf   &lt;==看看就好，不要眞的作！
[main]
cachedir=/var/cache/yum/$basearch/$releasever
keepcache=1
debuglevel=2
logfile=/var/log/yum.log
exactarch=1
obsoletes=1
....(底下省略)....
```

上述的特殊字体地方将 0 改成 1 ，这样就能够让你的 RPM 档案保存下来。不过，除非你有好多部主机要更新， 你想利用一台先 yum 升级且下载，然后将所有的 RPM 档案收集起来给内网的机器升级 (rpm -Fvh *.rpm) 之外， 上面的 vim 修改动作不建议修改！因为你的 /var 恐怕会被塞爆啊！再次提醒！

- yum 安装软件群组

什么是『软件群组』呢？由于 RPM 软件将一个大项目分成好几个小计划来执行，每个小计划都可以独立安装， 这样的好处是可以让使用者与软件开发者安装不同的环境！举例来说，在桌面系统中 (Desktop)，一般用户应该不会跑去发展软件吧？ 所以针对桌面计算机，软件群组又分为 "Desktop Platform" 与开发者 "Desktop Platform Development" 两部份， 每个软件群组内又含有多个不同的 RPM 软件档案！这样做的用途是方便使用者安装一整套的项目啦！

那么系统有多少软件群组呢？又该如何观察某个软件群组有拥有的 RPM 档案呢？我们就利用 Desktop Platform 这个项目来说明一下啰：

```
# 范例四：查询系统有的软件群组有多少个？
[root@www ~]# LANG=C yum grouplist
Installed Groups:             &lt;==这个是已安装的软件群组
   Additional Development
   Arabic Support
   Armenian Support
   Base
....(中间省略)....
Available Groups:             &lt;==这个是尚可安装的软件群组
   Afrikaans Support
   Albanian Support
   Amazigh Support
....(中间省略)....
 Desktop Platform
   Desktop Platform Development
....(后面省略)....

# 范例五：那么 Desktop Platform 内含多少个 RPM 软件呢？
[root@www ~]# yum groupinfo "Desktop Platform"
Group: 桌面环境平台
 Description: 受支援的 CentOS Linux 桌面平台函式库。
 Mandatory Packages: &lt;==主要的会被安装的软件有这些
   atk
....(中间省略)....
 Optional Packages:  &lt;==额外可选择的软件是这些
   qt-mysql
....(底下省略)....
# 如果你确定要安装这个软件群组的话，那就这样做：

[root@www ~]# yum groupinstall "Desktop Platform"
# 因为这里在介绍服务器的环境，所以上面的动作鸟哥是按下 n 来拒绝安装的！
```

利用这个『 yum groupinstall "软件群组名" 』可以让你一口气安装很多的软件， 而不必担心某个软件忘记装了！实在是很不错啦～而且利用 groupinfo 的功能你也可以发现一些不错的软件数据， 如此一来，你就可以更方便的管理你的 Linux 系统了，很不错吧！

- 全系统更新

我们都知道使用『yum update』就可以进行软件的更新。不过你晓得吗？ yum update 也可以直接进行同一版本的升级喔！举例来说，你可以从 6.0 升级到 6.1 版本哩！而且中间过程完全无痛哟！ 就跟一般软件升级而已，并没有不同哟！够愉快吧！

不过，如果你是想要从较旧版的 CentOS 5.x 升级到 6.x 的话，那么可能就得要多费些功夫了。为啥不要重灌比较快呢？因为你可能已经有些数据设定好，所以不想变更嘛！ 但老实说，不同版本 (ex> 5.x --> 6.x) 间的升级最好还是不要尝试啦！重新安装可能是最好的状况。底下列出酷学园的前辈提供的升级方式，以及 CentOS 官网直接提供的升级方式给你参考参考：

- 酷学园 TWU2 兄提供的 Red Hat 9 升级到 CentOS 3.x 的方法： http://phorum.study-area.org/index.php/topic,28648.html
- CentOS 官网提供的 CentOS 4.x 升级到 5.x 的方法： http://lists.centos.org/pipermail/centos-announce/2007-April/013660.html
- CentOS 维基百科提供的 CentOS 4.4 升级到 5.1 的方法： http://wiki.centos.org/HowTos/MigrationGuide/ServerCD_4.4_to_5

例题：请设定一下工作排程，让你的 CentOS 可以每天自动更新系统答：可以使用『 crontab -e 』来动作，也可以编辑『 vim /etc/crontab 』来动作， 由于这个更新是系统方面的，所以鸟哥习惯使用 vim /etc/crontab 来进行指令的说明。 其实内容很简单：

```
40 5 * * * root yum -y update && yum clean packages
```

这样就可以自动更新了， 时间订在每天的凌晨 5:40 。

7.2.4 挑选特定的映射站：修改 yum 配置文件与清除 yum 快取

虽然 yum 是你的主机能够联机上 Internet 就可以直接使用的，不过，由于 CentOS 的映射站台可能会选错， 举例来说，我们在台湾，但是 CentOS 的映射站台却选择到了大陆北京或者是日本去，有没有可能发生啊！ 有啊！鸟哥教学方面就常常发生这样的问题，要知道，我们联机到大陆或日本的速度是非常慢的呢！那怎办？ 当然就是手动的修改一下 yum 的配置文件就好啰！

在台湾，鸟哥熟悉的 CentOS 映射站台主要有昆山科大、高速网络中心与义守大学。 在学术网络之外，鸟哥近来比较偏好高速网络中心，似乎更新的速度比较快，而且连接台湾学术网络也非常快速哩！ 因此，鸟哥底下建议台湾的朋友使用高速网络中心的 ftp 主机资源来作为

yum 服务器来源喔！不过，因为鸟哥的机器很多都在昆山科大， 所以在学术网络上，使用的反而是昆山科大的 FTP 啰。目前高速网络中心对于 CentOS 所提供的相关网址如下：

- http://ftp.twaren.net/Linux/CentOS/6/

如果你连接到上述的网址后，就会发现里面有一堆连结，那些连结就是这个 yum 服务器所提供的容器了！ 所以高速网络中心也提供了 addons, centosplus, extras, fasttrack, os, updates 等容器，最好认的容器就是 os (系统默认的软件) 与 updates (软件升级版本) 啰！由于鸟哥在我的测试用主机是利用 x86_64 的版本， 因此那个 os 再点进去就会得到如下的可提供安装的网址：

- http://ftp.twaren.net/Linux/CentOS/6/os/x86_64/

为什么在上述的网址内呢？有什么特色！最重要的特色就是那个『 repodata 』的目录！该目录就是分析 RPM 软件后所产生的软件属性相依数据放置处！因此，当你要找容器所在网址时， 最重要的就是该网址底下一定要有个名为 repodata 的目录存在！那就是容器的网址了！其他的容器正确网址，就请各位看倌自行寻找一下喔！现在让我们修改配置文件吧！

```
[root@www ~]# vim /etc/yum.repos.d/CentOS-Base.repo
[base]
name=CentOS-$releasever - Base
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=os
#baseurl=http://mirror.centos.org/centos/$releasever/os/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6
```

如上所示，鸟哥仅列出 base 这个容器的原始内容而已，其他的容器内容请自行查阅啰！上面的数据需要注意的是：

- [base]： 代表容器的名字！中刮号一定要存在，里面的名称则可以随意取。但是不能有两个相同的容器名称，否则 yum 会不晓得该到哪里去找容器相关软件列表档案。
- name： 只是说明一下这个容器的意义而已，重要性不高！
- mirrorlist=： 列出这个容器可以使用的映射站台，如果不想使用，可以批注到这行。由于等一下我们是直接设定映像站， 因此这行待会儿确实是需要批注掉的喔！
- baseurl=： 这个最重要，因为后面接的就是容器的实际网址！ mirrorlist 是由 yum 程序自行去捉映像站台， baseurl 则是指定固定的一个容器网址！我们刚刚找到的网址放到这里来啦！
- enable=1： 就是让这个容器被启动。如果不想启动可以使用 enable=0 喔！
- gpgcheck=1： 还记得 RPM 的数字签名吗？这就是指定是否需要查阅 RPM 档案内的数字签名！
- gpgkey=：就是数字签名的公钥文件所在位置！使用默认值即可

了解这个配置文件之后，接下来让我们修改整个档案的内容，让我们这部主机可以直接使用高速网络中心的资源吧！ 修改的方式鸟哥仅列出 base 这个容器项目而已，其他的项目请您自行依照上述的作法来处理即可！

```
[root@www ~]# vim /etc/yum.repos.d/CentOS-Base.repo
[base]
name=CentOS-$releasever - Base
baseurl=http://ftp.twaren.net/Linux/CentOS/6/os/x86_64/   &lt;==就属它最重要！
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6
# 底下其他的容器项目，请自行到高速网络中心去查询后自己处理！

[root@www ~]# yum clean all  &lt;==改过配置文件，最好清除既有清单
```

接下来当然就是给他测试一下啰！如何测试呢？再次使用 yum 即可啊！

```
# 范例：列出目前 yum server 所使用的容器有哪些？
[root@www ~]# yum repolist all
repo id        repo name               status
base           CentOS-6 - Base         enabled: 6,019
c6-media       CentOS-6 - Media        disabled
centosplus     CentOS-6 - Plus         disabled
contrib        CentOS-6 - Contrib      disabled
debug          CentOS-6 - Debuginfo    disabled
extras         CentOS-6 - Extras       enabled:     0
updates        CentOS-6 - Updates      enabled: 1,042
repolist: 7,061
# 在 status 上写 enabled 才是有启动的！由于 /etc/yum.repos.d/
# 有多个配置文件，所以你会发现还有其他的容器存在。
```

- 修改容器产生的问题与解决之道

由于我们是修改系统默认的配置文件，事实上，我们应该要在 /etc/yum.repos.d/ 底下新建一个档案， 该扩展名必须是 .repo 才行！但因为我们使用的是指定特定的映射站台，而不是其他软件开发生提供的容器， 因此才修改系统默认配置文件。但是可能由于使用的容器版本有新旧之分，你得要知道， yum 会先下载容器的清单到本机的 /var/cache/yum 里面去！那我们修改了网址却没有修改容器名称 (中刮号内的文字)， 可能就会造成本机的列表与 yum 服务器的列表不同步，此时就会出现无法更新的问题了！

那怎么办啊？很简单，就清除掉本机上面的旧数据即可！需要手动处理吗？不需要的， 透过 yum 的 clean 项目来处理即可！

```
[root@www ~]# yum clean [packages&#124;headers&#124;all]
选项与参数：
 packages：将已下载的软件档案删除
 headers ：将下载的软件文件头删除
 all     ：将所有容器数据都删除！

# 范例：删除已下载过的所有容器的相关数据 (含软件本身与列表)
[root@www ~]# yum clean all
```

例题：有一个网址： http://free.nchc.org.tw/drbl-core/i386/RPMS.drbl-stable/ ，里面包含了台湾的国家高速网络中心所发展的自由软件。 请依据该网址提供的数据，做成系统可以自动网络安装的 yum 格式。答：由于 http://free.nchc.org.tw/drbl-core/i386/RPMS.drbl-stable/ 里面就有 repodata/ 目录，因此，这个网址可以直接做成 yum 的容器配置文件。 你可以这么做的：

```
[root@www ~]# vim /etc/yum.repos.d/drbl.repo
[drbl]
name=This is DRBL site
baseurl=http://free.nchc.org.tw/drbl-core/i386/RPMS.drbl-stable/
enable=1
gpgcheck=0

[root@www ~]# yum search drbl
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
============================== Matched: drbl ==============================
clonezilla.i386 : Opensource Clone System (ocs), clonezilla
drbl.i386 : DRBL (Diskless Remote Boot in Linux) package.
drbl-chntpw.i386 : Offline NT password and registry editor
....(底下省略)....

[root@www ~]# yum repolist all
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
repo id         repo name                status
base            CentOS-6 - Base          enabled: 6,019
c6-media        CentOS-6 - Media         disabled
centosplus      CentOS-6 - Plus          disabled
contrib         CentOS-6 - Contrib       disabled
debug           CentOS-6 - Debuginfo     disabled
drbl            This is DRBL site        enabled:    36 &lt;==新的在此！
extras          CentOS-6 - Extras        enabled:     0
updates         CentOS-6 - Updates       enabled: 1,042
repolist: 7,097
```

drbl 这个新增的容器里面，拥有 36 个软件喔！这样够清楚吗？

7.3 限制联机埠口 (port)

为什么我们的主机会响应网络上面的一些要求封包呢？例如我们设定了一部 WWW 主机后，当有来自 Internet 的 WWW 要求时，我们的主机就会予以回应，这是因为我们的主机有启用了 WWW 的监听埠口啊！所以，当我们启用了一个 daemon 时，就可能会造成主机的端口在进行监听的动作，此时该 daemon 就是已经对网络上面提供服务了！万一这个 daemon 程序有漏洞，因为他提供了 Internet 的服务，所以就容易被 Internet 上面的 cracker 所攻击了！所以说，仔细的检查自己系统上面的端口到底开了多少个，并且予以严格的管理，才能够降低被攻击的可能性啊！

7.3.1 什么是 port

快讲到烂掉了！当妳启动一个网络服务，这个服务会依据 TCP/IP 的相关通讯协议启动一个埠口在进行监听， 那就是 TCP/UDP 封包的 port (埠口) 了。我们从第二章也知道网络联机是双向的，服务器端得要启动一个监听的埠口， 客户端得要随机启动一个埠口来接收响应的数据才行。那么服务器端的服务是否需要启动在固定的埠口？ 客户端的埠口是否又是固定的呢？ 我们将第二章中与 port 有关的资料给她汇整一下先：

- 服务器端启动的监听埠口所对应的服务是固定的： 例如 WWW 服务开启在 port 80 ，FTP 服务开启在 port 21，email 传送开启在 port 25 等等，都是通讯协议上面的规范！
- 客户端启动程序时，随机启动一个大于 1024以上的埠口： 客户端启动的 port 是随机产生的，主要是开启在大于 1024 以上的埠口。这个 port 也是由某些软件所产生的， 例如浏览器、Filezilla 这个 FTP 客户端程序等等。
- 一部服务器可以同时提供多种服务： 所谓的『监听』是某个服务程序会一直常驻在内存当中，所以该程序启动的 port 就会一直存在。 只要服务器软件激活的埠口不同，那就不会造成冲突。当客户端连接到此服务器时，透过不同的埠口，就可以取得不同的服务数据啰。 所以，一部主机上面当然可以同时启动很多不同的服务啊！
- 共 65536 个 port： 由第二章的 TCP/UDP 表头数据中，就知道 port 占用 16 个位，因此一般主机会有 65536 个 port，而这些 port 又分成两个部分，以 port 1024 作区隔：
    - 只有 root 才能启动的保留的 port： 在小于 1024 的埠口，都是需要以 root 的身份才能启动的，这些 port 主要是用于一些常见的通讯服务，在 Linux 系统下，常见的协议与 port 的对应是记录在 /etc/services 里面的。
    - 大于 1024 用于 client 端的 port： 在大于 1024 以上的 port 主要是作为 client 端的软件激活的 port 。

- 是否需要三向交握： 建立可靠的联机服务需要使用到 TCP 协议，也就需要所谓的三向交握了，如果是非面向连接的服务，例如 DNS 与视讯系统， 那只要使用 UDP 协议即可。
- 通讯协议可以启用在非正规的 port： 我们知道浏览器默认会连接到 WWW 主机的 port 80，那么你的 WWW 是否可以启动在非 80 的其他埠口？ 当然可以啊！你可以透过 WWW 软件的设定功能将该软件使用的 port 启动在非正规的埠口， 只是如此一来，您的客户端要连接到你的主机时，就得要在浏览器的地方额外指定你所启用的非正规的埠口才行。 这个启动在非正规的端口功能，常常被用在一些所谓的地下网站啦！^_^。另外，某些软件默认就启动在大于 1024 以上的端口，如 MySQL 数据库软件就启动在 3306。
- 所谓的 port 的安全性： 事实上，没有所谓的 port 的安全性！因为『Port 的启用是由服务软件所造成的』， 也就是说，真正影响网络安全的并不是 port ，而是启动 port 的那个软件 (程序)！ 或许你偶而会听到：『没有修补过漏洞的 bind 8.x 版，很容易被黑客所入侵，请尽快升级到 bind 9.x 以后版本』，所以啰，对安全真正有危害的是『某些不安全的服务』 而不是『开了哪些 port 』才是！因此，没有必要的服务就将他关闭吧！ 尤其某些网络服务还会启动一些 port 哩！另外，那些已启动的软件也需要持续的保持更新喔！

7.3.2 埠口的观察： netstat, nmap

好了，我们现在知道这个 port 是什么鬼东西了，再来就是要来了解一下，我们的主机到底是开了多少的 port 呢？由于 port 的启动与服务有关，那么『服务』跟『 port 』对应的档案是哪一个？再提醒一次呦！是『 /etc/services 』啦！而常用来观察 port 的则有底下两个程序：

- netstat：在本机上面以自己的程序监测自己的 port；
- nmap：透过网络的侦测软件辅助，可侦测非本机上的其他网络主机，但有违法之虞。

见他的大头王！怎么使用 nmap 会违法？由于 nmap 的功能太强大了，所以很多 cracker 会直接以他来侦测别人的主机，这个时候就可能造成违法啦！只要你使用 nmap 的时候不要去侦测别人的计算机主机，那么就不会有问题啦！底下我们分别来说一说这两个宝贝吧！

- netstat

在做为服务器的 Linux 系统中，开启的网络服务越少越好！ 因为较少的服务可以较容易除错 (debug) 与了解安全漏洞，并可避免不必要的入侵管道！ 所以，这个时候请了解一下您的系统当中有没有哪些服务被开启了呢？ 要了解自己的系统当中的服务项目，最简便的方法就是使用 netstat 了！这个东西不但简单，而且功能也是很不错的。 这个指令的使用方法在 第五章常用网络功能指令介绍当中提过了， 底下我们仅提供如何使用这个工具的方法啰！

- 列出在监听的网络服务：

```
[root@www ~]# netstat -tunl
ctive Internet connections (only servers)
Proto Recv-Q Send-Q Local Address     Foreign Address     State
tcp        0      0 0.0.0.0:111       0.0.0.0:*           LISTEN
tcp        0      0 0.0.0.0:22        0.0.0.0:*           LISTEN
tcp        0      0 127.0.0.1:25      0.0.0.0:*           LISTEN
....(底下省略)....
```

上面说明了我的主机至少有启动 port 111, 22, 25 等，而且观察各联机接口，可发现 25 为 TCP 埠口，但只针对 lo 内部循环测试网络提供服务，因特网是连不到该埠口的。至于 port 22 则有提供因特网的联机功能。

- 列出已联机的网络联机状态：

```
[root@www ~]# netstat -tun
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address         Foreign Address      State
tcp        0     52 192.168.1.100:22     192.168.1.101:2162   ESTABLISHED
```

从上面的数据来看，我的本地端服务器 (Local Address, 192.168.1.100) 目前仅有一条已建立的联机，那就是与 192.168.1.101 那部主机连接的联机，并且联机方线是由对方连接到我主机的 port 22 来取用我服务器的服务呐！

- 删除已建立或在监听当中的联机：

如果想要将已经建立，或者是正在监听当中的网络服务关闭的话，最简单的方法当然就是找出该联机的 PID， 然后将他 kill 掉即可啊！例如下面的范例：

```
[root@www ~]# netstat -tunp
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address     Foreign Address     State       PID/P name
tcp        0     52 192.168.1.100:22 192.168.1.101:2162  ESTABLISHED 1342/0
```

如上面的范例，我们可以找出来该联机是由 sshd 这个程序来启用的，并且他的 PID 是 1342， 希望你不要心急的用 killall 这个指令，否则容易删错人 (因为你的主机里面可能会有多个 sshd 存在)， 应该要使用 kill 这个指令才对喔！

```
[root@www ~]# kill -9 1342
```

- nmap

如果你要侦测的设备并没有可让你登入的操作系统时，那该怎么办？举例来说，你想要了解一下公司的网络打印机是否有开放某些协议时， 那该如何处理啊？现在你知道 netstat 可以用来查阅本机上面的许多监听中的通讯协议， 那例如网络打印机这样的非本机的设备，要如何查询啊？呵呵！用 nmap 就对了！

nmap (注1)的软件说明之名称为：『Network exploration tool and security / port scanner』，顾名思义， 这个东西是被系统管理员用来管理系统安全性查核的工具！他的具体描述当中也提到了， nmap 可以经由程序内部自行定义的几个 port 对应的指纹数据，来查出该 port 的服务为何，所以我们也可以藉此了解我们主机的 port 到底是干嘛用的！在 CentOS 里头是有提供 nmap 的， 如果你没有安装，那么就使用 yum 去安装他吧！

```
[root@www ~]# nmap [扫瞄类型] [扫瞄参数] [hosts 地址与范围]
选项与参数：
[扫瞄类型]：主要的扫瞄类型有底下几种：
    -sT：扫瞄 TCP 封包已建立的联机 connect() ！
    -sS：扫瞄 TCP 封包带有 SYN 卷标的数据
    -sP：以 ping 的方式进行扫瞄
    -sU：以 UDP 的封包格式进行扫瞄
    -sO：以 IP 的协议 (protocol) 进行主机的扫瞄
[扫瞄参数]：主要的扫瞄参数有几种：
    -PT：使用 TCP 里头的 ping 的方式来进行扫瞄，可以获知目前有几部计算机存活(较常用)
    -PI：使用实际的 ping (带有 ICMP 封包的) 来进行扫瞄
    -p ：这个是 port range ，例如 1024-, 80-1023, 30000-60000 等等的使用方式
[Hosts 地址与范围]：这个有趣多了，有几种类似的类型
    192.168.1.100  ：直接写入 HOST IP 而已，仅检查一部；
    192.168.1.0/24 ：为 C Class 的型态，
    192.168.*.*    +   ：嘿嘿！则变为 B Class 的型态了！扫瞄的范围变广了！
    192.168.1.0-50,60-100,103,200 ：这种是变形的主机范围啦！很好用吧！

# 范例一：使用预设参数扫瞄本机所启用的 port (只会扫瞄 TCP)
[root@www ~]# yum install nmap
[root@www ~]# nmap localhost
PORT    STATE SERVICE
22/tcp  open  ssh
25/tcp  open  smtp
111/tcp open  rpcbind
# 在预设的情况下，nmap 仅会扫瞄 TCP 的协议喔！
```

nmap 的用法很简单吶！就直接在指令后面接上 IP 或者是主机名即可。不过，在预设的情况下 nmap 仅会帮你分析 TCP 这个通讯协议而已，像上面这个例子的输出结果。但优点是顺道也将开启该埠口的服务也列出来了， 真是好！ ^_^！那如果想要同时分析 TCP/UDP 这两个常见的通讯协议呢？可以这样做：

```
# 范例二：同时扫瞄本机的 TCP/UDP 埠口
[root@www ~]# nmap -sTU localhost
PORT    STATE SERVICE
22/tcp  open  ssh
25/tcp  open  smtp
111/tcp open  rpcbind
111/udp open  rpcbind  &lt;==会多出 UDP 的通讯协议埠口！
```

嘿嘿！与前面的范例比较一下，你会发现这次多了几个 UDP 的埠口，这样分析好多了！然后， 如果你想要了解一下到底有几部主机活在你的网络当中时，则可以这样做：

```
# 范例三：透过 ICMP 封包的检测，分析区网内有几部主机是启动的
[root@www ~]# nmap -sP 192.168.1.0/24
Starting Nmap 5.21 ( http://nmap.org ) at 2011-07-20 17:05 CST
Nmap scan report for www.centos.vbird (192.168.1.100)
Host is up.
Nmap scan report for 192.168.1.101 &lt;==这三行讲的是 192.168.101 的范例！
Host is up (0.00024s latency).
MAC Address: 00:1B:FC:58:9A:BB (Asustek Computer)
Nmap scan report for 192.168.1.254
Host is up (0.00026s latency).
MAC Address: 00:0C:6E:85:D5:69 (Asustek Computer)
Nmap done: 256 IP addresses (3 hosts up) scanned in 3.81 seconds
```

看到否？鸟哥的环境当中有三部主机活着呐 (Host is up)！并且该 IP 所对应的 MAC 也会被记录下来， 很不错吧！如果你还想要将各个主机的启动的 port 作一番侦测的话，那就得要使用：

```
[root@www ~]# nmap 192.168.1.0/24
```

之后你就会看到一堆 port number 被输出到屏幕上啰～如果想要随时记录整个网段的主机是否不小心开放了某些服务， 嘿嘿！利用 nmap 配合数据流重导向 (>, >> 等) 来输出成为档案， 那随时可以掌握住您局域网络内每部主机的服务启动状况啊！ ^_^

请特别留意，这个 nmap 的功能相当的强大，也是因为如此，所以很多刚在练习的黑客会使用这个软件来侦测别人的计算机。 这个时候请您特别留意，目前很多的人已经都有『特别的方式』来进行登录的工作！例如以 TCP_Wrappers (/etc/hosts.allow, /etc/hosts.deny) 的功能来记录曾经侦测过该 port 的 IP！ 这个软件用来『侦测自己机器的安全性』是很不错的一个工具，但是如果用来侦测别人的主机， 可是会『吃上官司』的！特别留意！！

7.3.3 埠口与服务的启动/关闭及开机时状态设定

从第二章的数据我们就知道，其实 port 是由执行某些软件之后被软件激活的。所以要关闭某些 port 时，那就直接将某个程序给他关闭就是了！关闭的方法你当然可以使用 kill，不过这毕竟不是正统的解决之道，因为 kill 这个指令通常具有强制关闭某些程序的功能，但我们想要正常的关闭该程序啊！ 所以，就利用系统给我们的 script 来关闭就好了啊。 在此同时，我们就得再来稍微复习一下，一般传统的服务有哪几种类型？

- stand alone 与 super daemon

我们在基础学习篇内谈到，在一般正常的 Linux 系统环境下，服务的启动与管理主要有两种方式：

- Stand alone 顾名思义，stand alone 就是直接执行该服务的执行档，让该执行文件直接加载到内存当中运作， 用这种方式来启动可以让该服务具有较快速响应的优点。一般来说，这种服务的启动 script 都会放置到 /etc/init.d/ 这个目录底下，所以你通常可以使用：『 /etc/init.d/sshd restart 』之类的方式来重新启动这种服务；
- Super daemon 用一个超级服务作为总管，来统一管理某些特殊的服务。在 CentOS 6.x 里面使用的则是 xinetd 这个 super daemon 啊！这种方式启动的网络服务虽然在响应上速度会比较慢， 不过，可以透过 super daemon 额外提供一些控管，例如控制何时启动、何时可以进行联机、 那个 IP 可以连进来、是否允许同时联机等等。通常个别服务的配置文件放置在 /etc/xinetd.d/ 当中，但设定完毕后需要重新以『 /etc/init.d/xinetd restart 』重新来启动才行！

关于更详细的服务说明，请参考基础篇的认识服务一文， 鸟哥在这里不再赘述。好，那么如果我想要将我系统上面的 port 111 关掉的话， 那应该如何关闭呢？最简单的作法就是先找出那个 port 111 的启动程序喔！

```
[root@www ~]# netstat -tnlp &#124; grep 111
tcp        0      0 0.0.0.0:111     0.0.0.0:*        LISTEN   990/rpcbind
tcp        0      0 :::111          :::*             LISTEN   990/rpcbind
# 原来用的是 rpcbind 这个服务程序！

[root@www ~]# which rpcbind
/sbin/rpcbind
# 找到档案后，再以 rpm 处理处理

[root@www ~]# rpm -qf /sbin/rpcbind
rpcbind-0.2.0-8.el6.x86_64
# 找到了！就是这个软件！所以将他关闭的方法可能就是：

[root@www ~]# rpm -qc rpcbind &#124; grep init
/etc/rc.d/init.d/rpcbind
[root@www ~]# /etc/init.d/rpcbind stop
```

透过上面的这个分析的流程，你可以利用系统提供的很多方便的工具来达成某个服务的关闭！ 为啥这么麻烦？不是利用 kill -9 990 就可以删掉该服务了吗？ 是没错啦！不过，你知道该服务是做啥用的吗？你知道将他关闭之后，你的系统会出什么问题吗？ 如果不知道的话，那么利用上面的流程不就可以找出该服务软件，再利用 rpm 查询功能， 不就能够知道该服务的作用了？所以说，这个方式还是对您会有帮助的啦！ 底下请您试着将您 CentOS 或者是其他版本的 Linux 的 Telnet 打开试看看。

例题：我们知道系统的 Telnet 服务通常是以 super daemon 来控管的，请您启动您系统的 telnet 试看看。答：

1. 要启动 telnet 首先必须要已经安装了 telnet 的服务器才行，所以请先以 rpm 查询看看是否有安装 telnet-server 呢？『rpm -qa | grep telnet-server』如果没有安装的话，请利用原版光盘来安装，或者使用『yum install telnet-server』 安装一下先；
2. 由于是 super daemon 控管，所以请编辑 /etc/xinetd.d/telnet 这个档案，将其中的『disable = yes』改成 『disable = no』之后以『/etc/init.d/xinetd restart』重新启动 super daemon 吧！

3. 利用 netstat -tnlp 察看是否有启动 port 23 呢？

- 预设启动的服务

刚刚上头的作法仅是『立即将该服务启动或关闭』喔！并不会影响到下次开机时，这个服务是否预设启动的情况。 如果你想要在开机的时候就启动或不启动某项服务时，那就得要了解一下基础学习篇里面谈到的开机流程管理的内容啦！在 Unix like 的系统当中我们都是透过 run level 来设定某些执行等级需要启动的服务，以 Red Hat 系统来说，这些 run level 启动的数据都是放置在 /etc/rc.d/rc[0-6].d/ 里面的，那如何管理该目录下的 script 呢？手动处理吗？会疯掉的呐！所以你必须要熟悉 chkconfig 或 Red Hat 系统的 ntsysv 这几个指令才行！

Tips: 这几个指令不熟吗？这个时候鸟哥不得不说了：『有 man 堪用直需用，莫待无 man 空自猜』赶紧给他 man 下去啦！



例题：(1)如何查阅 rpcbind 这个程序一开机就执行？ (2)如果开机就执行，如何将他改为开机时不要启动？ (3)如何立即关闭这个 rpcbind 服务？答：

1. 可以透过『 chkconfig --list | grep rpcbind 』与『 runlevel 』确认一下你的环境与 rpcbind 是否启动？
2. 如果有启动，可透过『 chkconfig --level 35 rpcbind off 』来设定开机时不要启动；
3. 可以透过『 /etc/init.d/rpcbind stop 』来立即关闭他！

聪明的你一定会问说：『鸟哥，你的意思是只要将系统所有的服务都关闭，那系统就会安全啰？』 当然....不是！因为『很多的系统服务是必须要存在的，否则系统将会出问题』 举例来说，那个保持系统可以具有工作排程的 crond 服务就一定要存在，而那个记录系统状况的 rsyslogd 也当然要存在～否则怎知道系统出了啥问题？所以啰，除非你知道每个服务的目的是啥，否则不要随便关闭该服务。 底下鸟哥列出几个常见的必须要存在的系统服务给大家参考参考先！这些服务请不要关闭啊！

服务名称	服务内容
acpid	新版的电源管理模块，通常建议开启，不过，某些笔记本电脑可能不支持此项服务，那就得关闭
atd	在管理单一预约命令执行的服务，应该要启动的
crond	在管理工作排程的重要服务，请务必要启动啊！
haldaemon	作系统硬件变更侦测的服务，与 USB 设备关系很大
iptables	Linux 内建的防火墙软件，这个也可以启动啦！
network	这个重要了吧？要网络就要有他啊！
postfix	系统内部邮件传递服务，不要随便关闭他！
rsyslog	系统的登录文件记录，很重要的，务必启动啊！
sshd	这是系统默认会启动的，可以让你在远程以文字型态的终端机登入喔！
xinetd	就是那个 super daemon 嘛！所以也要启动啦！

上面列出的是主机需要的重点服务，请您不要关闭他！除非你知道作了之后会有什么后果。举例来说，你如果不需要管理电源， 那么将 acpid 关闭也没有关系啊！如果你不需要提供远程联机功能，那么 sshd 也可以关闭啊！那其他你不知道的服务怎办？ 没关系，只要不是网络服务，你都可以保留他！如果是网络服务呢？那...鸟哥建议你不知道的服务就先关闭他！以后我们谈到每个相关的服务时，再一个一个打开即可。底下我们就来做作看关闭网络服务这个部分！

7.3.4 安全性考虑-关闭网络服务端口

我们的 Linux distribution 很好心的帮使用者想到很多了，所以在一安装完毕之后， 系统会开启一堆有的没有的网络服务，例如那个 rpcbind 之类的咚咚，这些东西你或许知道或许不知道，不过他就是有开启～ 但我们的主机明明就是用来做为服务器的，所以这些本来预计要给 client 使用的服务其实有点『多此一举』的感觉～ 所以啦，请你将他关闭吧！底下我们举个简单的例子来处理，将你的网络服务关闭就好，其他在系统内部的服务，就暂时保留吧！

例题：找出目前系统上面正在运作中的服务，并且找到相对应的启动脚本 (在 /etc/init.d 内的档名之意)。答：要找出服务，就利用 netstat -tunlp 即可找到！以鸟哥从第一章安装的示范机为例，鸟哥目前启动的网络服务有底下这些：

```
[root@www ~]# netstat -tlunp
Active Internet connections (only servers)
Proto  Local Address        State      PID/Program name
tcp    0.0.0.0:22           LISTEN     1176/sshd
tcp    127.0.0.1:25         LISTEN     1252/master
tcp    0.0.0.0:37753        LISTEN     1008/rpc.statd
tcp    :::22                LISTEN     1176/sshd
tcp    :::23                LISTEN     1851/xinetd
tcp    ::1:25               LISTEN     1252/master
tcp    :::38149             LISTEN     1008/rpc.statd
tcp    0.0.0.0:111          LISTEN     1873/rpcbind
tcp  0 :::111               LISTEN     1873/rpcbind
udp  0 0.0.0.0:111                     1873/rpcbind
udp  0 0.0.0.0:776                     1873/rpcbind
udp  0 :::111                          1873/rpcbind
udp  0 :::776                          1873/rpcbind
udp    0.0.0.0:760                     1008/rpc.statd
udp    0.0.0.0:52525                   1008/rpc.statd
udp    :::52343                        1008/rpc.statd
# 上述的输出鸟哥有稍微简化一些喔，所以有些字段不见了。
# 这个重点只是要展现出最后一个字段而已啦！
```

看起来总共有 sshd, master, rpc.statd, xinetd, rpcbind 等这几个服务，对照前一小节的数据内容来看， master (port 25), sshd 不能关掉，那么其他的就予以关闭啊！透过前两个小节的介绍，使用 which 与 rpm 搜寻吧！举例来说， rpc.statd 的启动脚本在：『rpm -qc $(rpm -qf $(which rpc.statd)) | grep init』这样找，结果是在『/etc/rc.d/init.d/nfslock』这里！ 因此最终的结果如下：

```
rpc.statd /etc/rc.d/init.d/nfs
          /etc/rc.d/init.d/nfslock
          /etc/rc.d/init.d/rpcgssd
          /etc/rc.d/init.d/rpcidmapd
          /etc/rc.d/init.d/rpcsvcgssd
xinetd    /etc/rc.d/init.d/xinetd
rpcbind   /etc/rc.d/init.d/rpcbind
```

接下来就是将该服务关闭，并且设定为开机不启动吧！

```
[root@www ~]# vim bin/closedaemon.sh
for daemon in nfs nfslock rpcgssd rpcidmapd rpcsvcgssd xinetd rpcbind
do
    chkconfig $daemon off
    /etc/init.d/$daemon stop
done
[root@www ~]# sh bin/closedaemon.sh
```

做完上面的例子之后，你再次下达 netstat -tlunp 之后，会得到仅剩 port 25, 22 而已！ 如此一来，绝大部分服务器用不到的服务就被你关闭，而且即使重新启动也不会被启动的啦！ ^_^

7.4 SELinux 管理原则

SELinux 使用所谓的委任式访问控制 (Mandatory Access Control, MAC) ，他可以针对特定的程序与特定的档案资源来进行权限的控管！ 也就是说，即使你是 root ，那么在使用不同的程序时，你所能取得的权限并不一定是 root ，而得要看当时该程序的设定而定。 如此一来，我们针对控制的『主体』变成了『程序』而不是『使用者』喔！因此，这个权限的管理模式就特别适合网络服务的『程序』了！ 因为，即使你的程序使用 root 的身份去启动，如果这个程序被攻击而被取得操作权，那该程序能作的事情还是有限的， 因为被 SELinux 限制住了能进行的工作了嘛！

举例来说， WWW 服务器软件的达成程序为 httpd 这支程序， 而默认情况下， httpd 仅能在 /var/www/ 这个目录底下存取档案，如果 httpd 这个程序想要到其他目录去存取数据时，除了规则设定要开放外，目标目录也得要设定成 httpd 可读取的模式 (type) 才行喔！限制非常多！ 所以，即使不小心 httpd 被 cracker 取得了控制权，他也无权浏览 /etc/shadow 等重要的配置文件喔！

7.4.1 SELinux 的运作模式

再次的重复说明一下，SELinux 是透过 MAC 的方式来控管程序，他控制的主体是程序， 而目标则是该程序能否读取的『档案资源』！所以先来说明一下这些咚咚的相关性啦！

- 主体 (Subject)： SELinux 主要想要管理的就是程序，因此你可以将『主体』跟本章谈到的 process 划上等号；
- 目标 (Object)： 主体程序能否存取的『目标资源』一般就是文件系统。因此这个目标项目可以等文件系统划上等号；
- 政策 (Policy)： 由于程序与档案数量庞大，因此 SELinux 会依据某些服务来制订基本的存取安全性政策。这些政策内还会有详细的规则 (rule) 来指定不同的服务开放某些资源的存取与否。在目前的 CentOS 6.x 里面仅有提供两个主要的政策如下，一般来说，使用预设的 target 政策即可。
    - targeted：针对网络服务限制较多，针对本机限制较少，是预设的政策；
    - mls：完整的 SELinux 限制，限制方面较为严格。
- 安全性本文 (security context)： 我们刚刚谈到了主体、目标与政策面，但是主体能不能存取目标除了要符合政策指定之外，主体与目标的安全性本文必须一致才能够顺利存取。 这个安全性本文 (security context) 有点类似文件系统的 rwx 啦！安全性本文的内容与设定是非常重要的！ 如果设定错误，你的某些服务(主体程序)就无法存取文件系统(目标资源)，当然就会一直出现『权限不符』的错误讯息了！

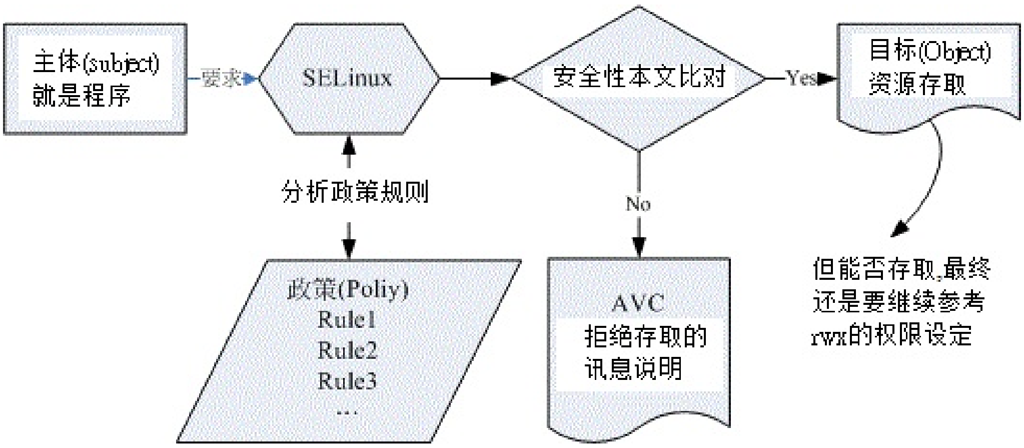


图 7.4-1、SELinux 运作的各组件之相关性(本图参考小州老师的上课讲义)

上图的重点在『主体』如何取得『目标』的资源访问权限！ 由上图我们可以发现，(1)主体程序必须要通过 SELinux 政策内的规则放行后，就可以与目标资源进行安全性本文的比对， (2)若比对失败则无法存取目标，若比对成功则可以开始存取目标。问题是，最终能否存取目标还是与文件系统的 rwx 权限设定有关喔！如此一来，加入了 SELinux 之后，出现权限不符的情况时，你就得要一步一步的分析可能的问题了！

- 安全性本文 (Security Context)

CentOS 6.x 的 target 政策已经帮我们制订好非常多的规则了，因此你只要知道如何开启/关闭某项规则的放行与否即可。 那个安全性本文比较麻烦！因为你可能需要自行配置文件案的安全性本文呢！为何需要自行设定啊？ 举例来说，你不也常常进行档案的 rwx 的重新设定吗？这个安全性本文你就将他想成 SELinux 内必备的 rwx 就是了！这样比较好理解啦。

安全性本文存在于主体程序中与目标档案资源中。程序在内存内，所以安全性本文可以存入是没问题。 那档案的安全性本文是记录在哪里呢？事实上，安全性本文是放置到档案的 inode 内的，因此主体程序想要读取目标档案资源时，同样需要读取 inode ， 这 inode 内就可以比对安全性本文以及 rwx 等权限值是否正确，而给予适当的读取权限依据。

那么安全性本文到底是什么样的存在呢？我们先来看看 /root 底下的档案的安全性本文好了。 观察安全性本文可使用『 ls -Z 』去观察如下：(注意：你必须已经启动了 SELinux 才行！若尚未启动，这部份请稍微看过一遍即可。底下会介绍如何启动 SELinux 喔！)

```
[root@www ~]# ls -Z
-rw-------. root  root  system_u:object_r:admin_home_t:s0     anaconda-ks.cfg
drwxr-xr-x. root  root  unconfined_u:object_r:admin_home_t:s0 bin
-rw-r--r--. root  root  system_u:object_r:admin_home_t:s0     install.log
-rw-r--r--. root  root  system_u:object_r:admin_home_t:s0     install.log.syslog
# 上述特殊字体的部分，就是安全性本文的内容！
```

如上所示，安全性本文主要用冒号分为三个字段 (最后一个字段先略过不看)，这三个字段的意义为：

```
Identify:role:type
身份识别:角色:类型
```

- 身份识别 (Identify)： 相当于账号方面的身份识别！主要的身份识别则有底下三种常见的类型：
    - root：表示 root 的账号身份，如同上面的表格显示的是 root 家目录下的数据啊！
    - system_u：表示系统程序方面的识别，通常就是程序啰；
    - user_u：代表的是一般使用者账号相关的身份。
- 角色 (Role)： 透过角色字段，我们可以知道这个数据是属于程序、档案资源还是代表使用者。一般的角色有：
    - object_r：代表的是档案或目录等档案资源，这应该是最常见的啰；
    - system_r：代表的就是程序啦！不过，一般使用者也会被指定成为 system_r 喔！
- 类型 (Type)： 在预设的 targeted 政策中， Identify 与 Role 字段基本上是不重要的！重要的在于这个类型 (type) 字段！ 基本上，一个主体程序能不能读取到这个档案资源，与类型字段有关！而类型字段在档案与程序的定义不太相同，分别是：
    - type：在档案资源 (Object) 上面称为类型 (Type)；
    - domain：在主体程序 (Subject) 则称为领域 (domain) 了！ domain 需要与 type 搭配，则该程序才能够顺利的读取档案资源啦！

- 程序与档案 SELinux type 字段的相关性

那么这三个字段如何利用呢？首先我们来瞧瞧主体程序在这三个字段的意义为何！透过身份识别与角色字段的定义， 我们可以约略知道某个程序所代表的意义喔！基本上，这些对应资料在 targeted 政策下的对应如下：

身份识别	角色	该对应在 **targeted** 的意义
root	system_r	代表供 root 账号登入时所取得的权限
system_u	system_r	由于为系统账号，因此是非交谈式的系统运作程序
user_u	system_r	一般可登入用户的程序啰！

但就如上所述，其实最重要的字段是类型字段，主体与目标之间是否具有可以读写的权限，与程序的 domain 及档案的 type 有关！这两者的关系我们可以使用达成 WWW 服务器功能的 httpd 这支程序与 /var/www/html 这个网页放置的目录来说明。 首先，看看这两个咚咚的安全性本文内容先：

```
[root@www ~]# yum install httpd
[root@www ~]# ll -Zd /usr/sbin/httpd /var/www/html
-rwxr-xr-x. root root system_u:object_r:httpd_exec_t:s0 /usr/sbin/httpd
drwxr-xr-x. root root system_u:object_r:httpd_sys_content_t:s0 /var/www/html
# 两者的角色字段都是 object_r ，代表都是档案！而 httpd 属于 httpd_exec_t 类型，
# /var/www/html 则属于 httpd_sys_content_t 这个类型！
```

httpd 属于 httpd_exec_t 这个可以执行的类型，而 /var/www/html 则属于 httpd_sys_content_t 这个可以让 httpd 领域 (domain) 读取的类型。文字看起来不太容易了解吧！我们使用图示来说明这两者的关系！

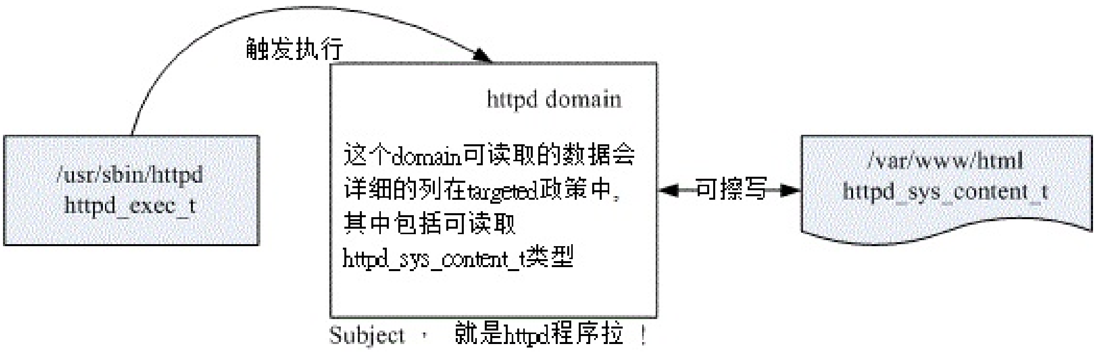


图 7.4-2、主体程序取得的 domain 与目标档案资源的 type 相互关系

上图的意义我们可以这样看的：

1. 首先，我们触发一个可执行的目标档案，那就是具有 httpd_exec_t 这个类型的 /usr/sbin/httpd
2. 该档案的类型会让这个档案所造成的主体程序 (Subject) 具有 httpd 这个领域 (domain)，我们的政策针对这个领域已经制定了许多规则，其中包括这个领域可以读取的目标资源类型；
3. 由于 httpd domain 被设定为可以读取 httpd_sys_content_t 这个类型的目标档案 (Object)， 因此你的网页放置到 /var/www/html/ 目录下，就能够被 httpd 那支程序所读取了；
4. 但最终能不能读到正确的资料，还得要看 rwx 是否符合 Linux 权限的规范！

上述的流程告诉我们几个重点，第一个是政策内需要制订详细的 domain/type 相关性；第二个是若档案的 type 设定错误， 那么即使权限设定为 rwx 全开的 777 ，该主体程序也无法读取目标档案资源的啦！不过如此一来， 也就可以避免用户将他的家目录设定为 777 时所造成的权限困扰。

7.4.2 SELinux 的启动、关闭与观察

并非所有的 Linux distributions 都支持 SELinux 的，所以你必须要先观察一下你的系统版本为何！ 鸟哥这里介绍的 CentOS 6.x 本身就有支持 SELinux 啦！所以你不需要自行编译 SELinux 到你的 Linux 核心中！ 目前 SELinux 支持三种模式，分别如下：

- enforcing：强制模式，代表 SELinux 运作中，且已经正确的开始限制 domain/type 了；
- permissive：宽容模式：代表 SELinux 运作中，不过仅会有警告讯息并不会实际限制 domain/type 的存取。这种模式可以运来作为 SELinux 的 debug 之用；
- disabled：关闭，SELinux 并没有实际运作。

那你怎么知道目前的 SELinux 模式呢？就透过 getenforce 吧！

```
[root@www ~]# getenforce
Enforcing  &lt;==诺！就显示出目前的模式为 Enforcing 啰！
```

另外，我们又如何知道 SELinux 的政策 (Policy) 为何呢？这时可以来观察配置文件啦：

```
[root@www ~]# vim /etc/selinux/config
SELINUX=enforcing     &lt;==调整 enforcing&#124;disabled&#124;permissive
SELINUXTYPE=targeted  &lt;==目前仅有 targeted 与 mls
```

- SELinux 的启动与关闭

上面是默认的政策与启动的模式！你要注意的是，如果改变了政策则需要重新启动；如果由 enforcing 或 permissive 改成 disabled ，或由 disabled 改成其他两个，那也必须要重新启动。这是因为 SELinux 是整合到核心里面去的， 你只可以在 SELinux 运作下切换成为强制 (enforcing) 或宽容 (permissive) 模式，不能够直接关闭 SELinux 的！ 如果刚刚你发现 getenforce 出现 disabled 时，请到上述档案修改成为 enforcing 然后重新启动吧！

不过你要注意的是，如果从 disable 转到启动 SELinux 的模式时， 由于系统必须要针对档案写入安全性本文的信息，因此开机过程会花费不少时间在等待重新写入 SELinux 安全性本文 (有时也称为 SELinux Label) ，而且在写完之后还得要再次的重新启动一次喔！你必须要等待粉长一段时间！ 等到下次开机成功后，再使用 getenforce 来观察看看有否成功的启动到 Enforcing 的模式啰！

如果你已经在 Enforcing 的模式，但是可能由于一些设定的问题导致 SELinux 让某些服务无法正常的运作， 此时你可以将 Enforcing 的模式改为宽容 (permissive) 的模式，让 SELinux 只会警告无法顺利联机的讯息， 而不是直接抵挡主体程序的读取权限。让 SELinux 模式在 enforcing 与 permissive 之间切换的方法为：

```
[root@www ~]# setenforce [0&#124;1]
选项与参数：
0 ：转成 permissive 宽容模式；
1 ：转成 Enforcing 强制模式

# 范例一：将 SELinux 在 Enforcing 与 permissive 之间切换与观察
[root@www ~]# setenforce 0
[root@www ~]# getenforce
Permissive
[root@www ~]# setenforce 1
[root@www ~]# getenforce
Enforcing
```

不过请注意， setenforce 无法在 Disabled 的模式底下进行模式的切换喔！

Tips: 在某些特殊的情况底下，你从 Disabled 切换成 Enforcing 之后，竟然有一堆服务无法顺利启动，都会跟你说在 /lib/xxx 里面的数据没有权限读取，所以启动失败。这大多是由于在重新写入 SELinux type (Relable) 出错之故，使用 Permissive 就没有这个错误。那如何处理呢？最简单的方法就是在 Permissive 的状态下，使用『 restorecon -Rv / 』重新还原所有 SELinux 的类型，就能够处理这个错误！



7.4.3 SELinux type 的修改

既然 SELinux 的类型字段 (type) 这么重要，那如何修改与变更这个字段，当然就是最重要的一件事啰。 首先，我们来看看如果复制一个档案到不同的目录去，会发生什么状况吧！

```
# 范例：将 /etc/hosts 复制到 root 家目录，并观察相关的 SELinux 类型变化
[root@www ~]# cp /etc/hosts /root
[root@www ~]# ls -dZ /etc/hosts /root/hosts /root
-rw-r--r--. root root system_u:object_r:net_conf_t:s0  /etc/hosts
dr-xr-x---. root root system_u:object_r:admin_home_t:s0 /root
-rw-r--r--. root root unconfined_u:object_r:admin_home_t:s0 /root/hosts

# 范例：将 /root/hosts 移动到 /tmp 下，并观察相关的 SELinux 类型变化
[root@www ~]# mv /root/hosts /tmp
[root@www ~]# ls -dZ /tmp /tmp/hosts
drwxrwxrwt. root root system_u:object_r:tmp_t:s0       /tmp
-rw-r--r--. root root unconfined_u:object_r:admin_home_t:s0 /tmp/hosts
```

看到没有？当你单纯的复制时，SELinux 的 type 字段是会继承目标目录的，所以 /root/hosts 的类型就会变成 admin_home_t 这个类型了。但是如果是移动呢？那么连同 SELinux 的类型也会被移动过去，因此 /tmp/hosts 会依旧保持 admin_home_t 而不会变成 /tmp 的 tmp_t 这个类型呦！要注意！要注意！那么，如何将 /tmp/hosts 变更成为最原始的 net_conf_t 这个类型呢？那就得要使用 chcon 啰！

* chcon

```
[root@www ~]# chcon [-R] [-t type] [-u user] [-r role] 档案
[root@www ~]# chcon [-R] --reference=范例文件 档案
选项与参数：
-R  ：连同该目录下的次目录也同时修改；
-t  ：后面接安全性本文的类型字段！例如 httpd_sys_content_t ；
-u  ：后面接身份识别，例如 system_u；
-r  ：后面街角色，例如 system_r；
--reference=范例文件：拿某个档案当范例来修改后续接的档案的类型！

# 范例：将刚刚的 /tmp/hosts 类型改为 etc_t 的类型
[root@www ~]# chcon -t net_conf_t /tmp/hosts
[root@www ~]# ll -Z /tmp/hosts
-rw-r--r--. root root unconfined_u:object_r:net_conf_t:s0 /tmp/hosts

# 范例：以 /var/spool/mail/ 为依据，将 /tmp/hosts 修改成该类型
[root@www ~]# ll -dZ /var/spool/mail
drwxrwxr-x. root mail system_u:object_r:mail_spool_t:s0 /var/spool/mail
[root@www ~]# chcon --reference=/var/spool/mail /tmp/hosts
[root@www ~]# ll -Z /tmp/hosts
-rw-r--r--. root root system_u:object_r:mail_spool_t:s0 /tmp/hosts
```

chcon 的修改方式中，我们必须要知道最终我们的 SELinux type 是啥类型后，才能够变更成功。 如果你想要作的是『复原成原有的 SELinux type』呢？那可以参考底下的指令来进行吻！

- restorecon

```
[root@www ~]# restorecon [-Rv] 档案或目录
选项与参数：
-R  ：连同次目录一起修改；
-v  ：将过程显示到屏幕上

# 范例：将刚刚 /tmp/hosts 移动至 /root 并以预设的安全性本文改正过来
[root@www ~]# mv /tmp/hosts /root
[root@www ~]# ll -Z /root/hosts
-rw-r--r--. root root system_u:object_r:mail_spool_t:s0 /root/hosts
[root@www ~]# restorecon -Rv /root
restorecon reset /root/hosts context system_u:object_r:mail_spool_t:s0-&gt;
system_u:object_r:admin_home_t:s0
# 上面这两行其实是同一行喔！表示将 hosts 由 mail_spool_t 改为 admin_home_t
```

- 默认目录的安全性本文查询与修改

透过上面这几个练习，你就会知道啦，SELinux type 恐怕会在档案的复制/移动时产生一些变化，因此需要善用 chcon, restorecon 等指令来进行修订。那你应该还是会想到一件事，那就是， restorecon 怎么会知道每个目录记载的默认 SELinux type 类型呢？这是因为系统有记录嘛！记录在 /etc/selinux/targeted/contexts，但是该目录内有很多不同的数据， 要使用文本编辑器去查阅很麻烦，此时，我们可以透过 semanage 这个指令的功能来查询与修改喔！

```
[root@www ~]# semanage {login&#124;user&#124;port&#124;interface&#124;fcontext&#124;trans
[root@www ~]# semanage fcontext -{a&#124;d&#124;m} [-frst] file_spec
选项与参数：
fcontext ：主要用在安全性本文方面的用途， -l 为查询的意思；
-a ：增加的意思，你可以增加一些目录的默认安全性本文类型设定；
-m ：修改的意思；
-d ：删除的意思。

# 范例：查询一下 /var/www/ 的预设安全性本文设定为何！
[root@www ~]# yum install policycoreutils-python
[root@www ~]# semanage fcontext -l &#124; grep '/var/www'
SELinux fcontext            类型            Context
/var/www(/.*)?              all files     system_u:object_r:httpd_sys_content_t:s0
/var/www(/.*)?/logs(/.*)?  all files     system_u:object_r:httpd_log_t:s0
....(后面省略)....
```

从上面的说明，我们知道其实 semanage 可以处理非常多的任务，不过，在这个小节我们主要想了解的是每个目录的默认安全性本文。 如上面范例所示，我们可以查询的到每个目录的安全性本文啦！而目录的设定可以使用正规表示法去指定一个范围。那么如果我们想要增加某些自定义的目录的安全性本文呢？ 举例来说，我想要制订 /srv/vbird 成为 public_content_t 的类型时，应该如何指定呢？

```
# 范例：利用 semanage 设定 /srv/vbird 目录的默认安全性本文为 public_content_t
[root@www ~]# mkdir /srv/vbird
[root@www ~]# ll -Zd /srv/vbird
drwxr-xr-x. root root unconfined_u:object_r:var_t:s0   /srv/vbird
# 如上所示，预设的情况应该是 var_t 这个咚咚的！

[root@www ~]# semanage fcontext -l &#124; grep '/srv'
/srv                   directory    system_u:object_r:var_t:s0 &lt;==看这里
/srv/.*                all files    system_u:object_r:var_t:s0
....(底下省略)....
# 上面则是预设的 /srv 底下的安全性本文数据，不过，并没有指定到 /srv/vbird 啦

[root@www ~]# semanage fcontext -a -t public_content_t "/srv/vbird(/.*)?"
[root@www ~]# semanage fcontext -l &#124; grep '/srv/vbird'
/srv/vbird(/.*)?           all files   system_u:object_r:public_content_t:s0

[root@www ~]# cat /etc/selinux/targeted/contexts/files/file_contexts.local
# This file is auto-generated by libsemanage
# Please use the semanage command to make changes
/srv/vbird(/.*)?    system_u:object_r:public_content_t:s0
# 其实就是写入这个档案的啰！ ^_^

[root@www ~]# restorecon -Rv /srv/vbird* &lt;==尝试恢复默认值
[root@www ~]# ll -Zd /srv/vbird
drwxr-xr-x. root root system_u:object_r:public_content_t:s0 /srv/vbird
# 有默认值，以后用 restorecon 来修改比较简单！
```

semanage 的功能很多，不过鸟哥主要用到的仅有 fcontext 这个项目的动作而已。如上所示， 你可以使用 semanage 来查询所有的目录默认值，也能够使用他来增加默认值的设定！ 如果您学会这些基础的工具， 那么 SELinux 对你来说，也不是什么太难的咚咚啰！

7.4.4 SELinux 政策内的规则布尔值修订

前面讲到，要通过 SELinux 的验证之后才能开始档案权限 rwx 的判断，而 SELinux 的判断主要是 (1)政策内的规则比对与 (2)程序与档案的 SELinux type 要符合才能够放行。前一个小节谈的是 SELinux 的 type ，这个小节就是要谈一下政策内的规则啰， 包括如何查询与修改相关的规则放行与否啰。

- 政策查阅

CentOS 6.x 预设使使用 targeted 政策，那么这个政策提供多少相关的规则呢？此时可以透过 seinfo 来查询喔！

```
[root@www ~]# yum install setools-console
[root@www ~]# seinfo [-Atrub]
选项与参数：
-A  ：列出 SELinux 的状态、规则布尔值、身份识别、角色、类别等所有信息
-t  ：列出 SELinux 的所有类别 (type) 种类
-r  ：列出 SELinux 的所有角色 (role) 种类
-u  ：列出 SELinux 的所有身份识别 (user) 种类
-b  ：列出所有规则的种类 (布尔值)

# 范例一：列出 SELinux 在此政策下的统计状态
[root@www ~]# seinfo
tatistics for policy file: /etc/selinux/targeted/policy/policy.24
Policy Version & Type: v.24 (binary, mls)  &lt;==列出政策所在档与版本

   Classes:            77    Permissions:       229
   Sensitivities:       1    Categories:       1024
   Types:            3076    Attributes:        251
   Users:               9    Roles:              13
   Booleans:          173    Cond. Expr.:       208
   Allow:          271307    Neverallow:          0
   Auditallow:         44    Dontaudit:      163738
   Type_trans:      10941    Type_change:        38
   Type_member:        44    Role allow:         20
   Role_trans:        241    Range_trans:      2590
....(底下省略)....
# 从上面我们可以看到这个政策是 targeted ，此政策的 SELinux type 有 3076 个；
# 而针对网络服务的规则 (Booleans) 共制订了 173 条规则！

# 范例二：列出与 httpd 有关的规则 (booleans) 有哪些？
[root@www ~]# seinfo -b &#124; grep httpd
Conditional Booleans: 173
   allow_httpd_mod_auth_pam
   httpd_setrlimit
   httpd_enable_ftp_server
....(底下省略)....
# 你可以看到，有非常多的与 httpd 有关的规则订定呢！
```

从上面我们可以看到与 httpd 有关的布尔值，同样的，如果你想要找到有 httpd 字样的安全性本文类别时， 就可以使用『 seinfo -t | grep httpd 』来查询了！如果查询到相关的类别或者是布尔值后，想要知道详细的规则时， 就得要使用 sesearch 这个指令了！

```
[root@www ~]# sesearch [--all] [-s 主体类别] [-t 目标类别] [-b 布尔值]
选项与参数：
--all  ：列出该类别或布尔值的所有相关信息
-t   ：后面还要接类别，例如 -t httpd_t
-b   ：后面还要接布尔值的规则，例如 -b httpd_enable_ftp_server

# 范例一：找出目标档案资源类别为 httpd_sys_content_t 的有关信息
[root@www ~]# sesearch --all -t httpd_sys_content_t
Found 683 semantic av rules:
   allow avahi_t file_type : filesystem getattr ;
   allow corosync_t file_type : filesystem getattr ;
   allow munin_system_plugin_t file_type : filesystem getattr ;
....(底下省略)....
# 『 allow  主体程序安全性本文类别  目标档案安全性本文类别 』
# 如上，说明这个类别可以被那个主题程序的类别所读取，以及目标档案资源的格式。
```

你可以很轻易的查询到某个主体程序 (subject) 可以读取的目标档案资源 (Object)。 那如果是布尔值呢？里面又规范了什么？让我们来看看先：

```
# 范例三：我知道有个布尔值为 httpd_enable_homedirs ，请问该布尔值规范多少规则？
[root@www ~]# sesearch -b httpd_enable_homedirs --all
Found 43 semantic av rules:
   allow httpd_user_script_t user_home_dir_t : dir { getattr search open } ;
   allow httpd_sys_script_t user_home_dir_t : dir { ioctl read getattr  } ;
....(后面省略)....
```

从这个布尔值的设定我们可以看到里面规范了非常多的主体程序与目标档案资源的放行与否！ 所以你知道了，实际规范这些规则的，就是布尔值的项目啦！那也就是我们之前所说的一堆规则是也！ 你的主体程序能否对某些目标档案进行存取，与这个布尔值非常有关系喔！因为布尔值可以将规则设定为启动 (1) 或者是关闭 (0) 啦！

- 布尔值的查询与修改

上面我们透过 sesearch 知道了，其实 Subject 与 Object 能否有存取的权限，是与布尔值有关的， 那么系统有多少布尔值可以透过 seinfo -b 来查询，但，每个布尔值是启动的还是关闭的呢？这就来查询看看吧：

```
[root@www ~]# getsebool [-a] [布尔值条款]
选项与参数：
-a  ：列出目前系统上面的所有布尔值条款设定为开启或关闭值

# 范例一：查询本系统内所有的布尔值设定状况
[root@www ~]# getsebool -a
abrt_anon_write --&gt; off
allow_console_login --&gt; on
allow_cvs_read_shadow --&gt; off
....(底下省略)....
# 您瞧！这就告诉你目前的布尔值状态啰！
```

那么如果查询到某个布尔值，并且以 sesearch 知道该布尔值的用途后，想要关闭或启动他，又该如何处置？

```
[root@www ~]# setsebool [-P] 布尔值=[0&#124;1]
选项与参数：
-P  ：直接将设定值写入配置文件，该设定数据未来会生效的！

# 范例一：查询 httpd_enable_homedirs 是否为 on，若不为 on 请启动他！
[root@www ~]# getsebool httpd_enable_homedirs
httpd_enable_homedirs --&gt; off  &lt;==结果是 off ，依题意给他启动！

[root@www ~]# setsebool -P httpd_enable_homedirs=1
[root@www ~]# getsebool httpd_enable_homedirs
httpd_enable_homedirs --&gt; on
```

这个 setsebool 最好记得一定要加上 -P 的选项！因为这样才能将此设定写入配置文件！ 这是非常棒的工具组！你一定要知道如何使用 getsebool 与 setsebool 才行！

7.4.5 SELinux 登录文件记录所需服务

上述的指令功能当中，尤其是 setsebool, chcon, restorecon 等，都是为了当你的某些网络服务无法正常提供相关功能时， 才需要进行修改的一些指令动作。但是，我们怎么知道哪个时候才需要进行这些指令的修改啊？我们怎么知道系统因为 SELinux 的问题导致网络服务不对劲啊？如果都要靠客户端联机失败才来哭诉，那也太没有效率了！所以，我们的 CentOS 6.x 有提供几支侦测的服务在登录 SELinux 产生的错误喔！那就是 auditd 与 setroubleshootd。

- setroubleshoot --> 错误讯息写入 /var/log/messages

几乎所有 SELinux 相关的程序都会以 se 为开头，这个服务也是以 se 为开头！而 troubleshoot 大家都知道是错误克服， 因此这个 setroubleshoot 自然就得要启动他啦！这个服务会将关于 SELinux 的错误讯息与克服方法记录到 /var/log/messages 与 /var/log/setroubleshoot/* 里头，所以你一定得要启动这个服务才好。启动这个服务之前当然就是得要安装它啦！ 这玩意儿总共需要两个软件，分别是 setroublshoot 与 setroubleshoot-server，如果你没有安装，请自行使用 yum 安装吧！

此外，原本的 SELinux 信息本来是以两个服务来记录的，分别是 auditd 与 setroubleshootd。既然是同样的信息， 因此 CentOS 6.x 将两者整合在 auditd 当中啦！所以，并没有 setroubleshootd 的服务存在了喔！因此， 当你安装好了 setroubleshoot-server 之后，请记得要重新启动 auditd，否则 setroubleshootd 的功能不会被启动的。

```
[root@www ~]# yum install setroubleshoot setroubleshoot-server
[root@www ~]# /etc/init.d/auditd restart &lt;==整合到 auditd 当中了！
```

Tips: 事实上，CentOS 6.x 对 setroubleshootd 的运作方式是： (1)先由 auditd 去呼叫 audispd 服务， (2)然后 audispd 服务去启动 sedispatch 程序， (3)sedispatch 再将原本的 auditd 讯息转成 setroubleshootd 的讯息，进一步储存下来的！



那么如果有发生错误时，讯息像什么呢？我们使用 httpd 这支程序产生的错误来说明好了。假设你需要启动 WWW 服务器， 我们的 WWW 是由 httpd 这支服务提供的，因此你必须要安装且启动它才行：

```
[root@www ~]# /etc/init.d/httpd start
[root@www ~]# netstat -tlnp &#124; grep http
tcp      0    0 :::80    :::*             LISTEN      2218/httpd
# 看到没？有启动 port 80 了！这是重点！
```

这个时候我们的 WWW 服务器就安装妥当了。我们的首页其实是放置到 /var/www/html 目录下的，且文件名必须要是 index.html。 那如果我使用底下的模式来进行首页的处理时，可能就会产生 SELinux 的问题了！我们就来模拟一下出问题的状况吧！

```
[root@www ~]# echo "My first selinux check" &gt; index.html
[root@www ~]# ll index.html
-rw-r--r--. 1 root root 23 2011-07-20 18:16 index.html  &lt;==权限没问题
[root@www ~]# mv index.html /var/www/html
```

此时我们就可以打开浏览器，然后在浏览器上面输入 Linux 自己的 IP 来查察看，看能不能连上自己的 WWW 首页。 因为我们这次安装并没有图形接口，所以使用 links 来查察 http://localhost/index.html 看看！你会得到如下的讯息：

```
[root@www ~]# links http://localhost/index.html -dump
                                   Forbidden

   You don't have permission to access /index.html on this server.

   --------------------------------------------------------------------------

    Apache/2.2.15 (CentOS) Server at localhost Port 80
```

画面最明显的地方就是告诉你，你并没有权限可以存取 index.html 的！见鬼了！明明权限是对的喔！那怎办？ 没关系，就透过 setroubleshoot 的功能去检查看看。此时请分析一下 /var/log/messages 的内容吧！有点像这样：

```
[root@www ~]# cat /var/log/messages &#124; grep setroubleshoot
Jul 21 14:53:20 www setroubleshoot: SELinux is preventing /usr/sbin/httpd
"getattr" access to /var/www/html/index.html. For complete SELinux messages.
run sealert -l 6c927892-2469-4fcc-8568-949da0b4cf8d
```

上面的错误讯息可是同一行喔！大纲说的是『SElinux 被用来避免 httpd 读取到错误的安全性本文， 想要查阅完整的数据，请执行 sealert -l ...』没错！你注意到了！重点就是 sealert -l 啦！ 上面提供的信息并不完整，想要更完整的说明得要靠 sealert 配合侦测到的错误代码来处

理。 实际处理后会像这样：

```
[root@www ~]# sealert -l 6c927892-2469-4fcc-8568-949da0b4cf8d
Summary:

SELinux is preventing /usr/sbin/httpd "getattr" access to
/var/www/html/index.html.   &lt;==刚刚在 messages 里面看到的信息！

Detailed Description:       &lt;==接下来是详细的状况解析！要看喔！

SELinux denied access requested by httpd. /var/www/html/index.html may
be a mislabeled. /var/www/html/index.html default SELinux type is
httpd_sys_content_t, but its current type is admin_home_t. Changing
this file back to the default type, may fix your problem.
....(中间省略)....

Allowing Access:  &lt;==超重要的项目！要看要看！

You can restore the default system context to this file by executing the
restorecon command. restorecon '/var/www/html/index.html', if this file
is a directory, you can recursively restore using restorecon -R
'/var/www/html/index.html'.

Fix Command:

/sbin/restorecon '/var/www/html/index.html'  &lt;==知道如何解决了吗？

Additional Information:  &lt;==还有一些额外的信息！
....(底下省略)....

[root@www ~]# restorecon -Rv '/var/www/html/index.html'
restorecon reset /var/www/html/index.html context unconfined_u:object_r:
admin_home_t:s0-&gt;system_u:object_r:httpd_sys_content_t:s0
```

重点就是上面特殊字体显示的地方！你只要照着『Allowing Access』里面的提示去进行处理， 就能够完成你的 SELinux 类型设定了！比对刚刚我们上个小节提到的 restorecon 与 chcon 你就能够知道， setroubleshoot 提供的讯息有多有效了吧！不管出了啥 SELinux 的问题，绝大部分在 setroubleshoot 的服务中就会告诉你解决之道！所以，很多东西都不用背的！

- 用 email 或在指令列上面直接提供 setroubleshoot 错误讯息

如果每次测试都得要到 /var/log/messages 去分析，那真是挺麻烦的啊！没关系，我们可以透过 email 或 console 的方式来将信息产生！也就是说，我们可以让 setroubleshoot 主动的发送产生的信息到我们指定的 email，这样可以方便我们实时的分析喔！怎么办到？就修改 setroubleshoot 的配置文件即可。你可以查阅 /etc/setroubleshoot/setroubleshoot.cfg 这个档案的内容，我们只需要修改的地方如下：

```
[root@www ~]# vim /etc/setroubleshoot/setroubleshoot.cfg
[email]
# 大约在 81 行左右，这行要存在才行！
recipients_filepath = /var/lib/setroubleshoot/email_alert_recipients

# 大约在 147 行左右，将原本的 False 修改成 True 先！
console = True

[root@www ~]# vim /var/lib/setroubleshoot/email_alert_recipients
root@localhost
your@email.address

[root@www ~]# /etc/init.d/auditd restart
```

之后你就可以透过分析你的 email 来取得 SELinux 的错误讯息啰！非常的简单吧！只是要注意，上述的填写 email 的档案中， 不能只写账号，你要连同 @localhost 都写上，这样本机上面的 root 才能收到信件喔！就这么简单哩！ ^_^

- SELinux 错误克服的总结

我们来简单的做个总结吧！因为你的网络联机要通过 SELinux 才的权限判定后才能够继续 rwx 的权限比对。而 SELinux 的比对主要又分为： (1)需要通过政策的各项规则比对后 (2)才能够进行 SELinux type 安全性本文的比对，这两项工作都得要正确才行。而后续的 SELinux 修改主要是透过 chcon, restorecon, setsebool 等指令来处理的。但是如何处理呢？可以透过分析 /var/log/messages 内提供的 setroubleshoot 的信息来处置！这样就很轻松的可以管理你的 SELinux 啰！

但是如果因为某些原因，举例来说 CentOS 没有规范到的 setroubleshoot 信息时，可能你还是无法了解到事情到底是哪里出错。 那此时我们会这样建议：

1. 在服务与 rwx 权限都没有问题，却无法成功的使用网络服务时；
2. 先使用 setenforce 0 设定为宽容模式；
3. 再次使用该网络服务，如果这样就能用，表示 SELinux 出问题，请往下继续处理。如果这样还不能用，那问题就不是在 SELinux 上面！请再找其他解决方法，底下的动作不适合你；
4. 分析 /var/log/messages 内的信息，找到 sealert -l 相关的信息并且执行；
5. 找到 Allow Access 的关键词，照里面的动作来进行 SELinux 的错误克服；
6. 处理完毕重新 setenforce 1 ，再次测试网络服务吧！

这样就能够很轻松的管理你的 SELinux 啦！不需要想太多！分析登录档就对啦！

Tips: 当鸟哥第一次修改这个 SELinux 的部分时，在 sealert 的部分一直出现错误，信息为：query_alert error (1003)... 后来经过更新软件后，又发现无法以 UTF8 进行文字译码的问题！实在伤脑筋～最后还是修改了 /etc/sysconfig/i18n 将里面的数据设定为： LANG=en_US 并且重新启动，才顺利恢复 sealert 的信息说明！真的是很怪异！


鸟哥
VBird

7.5 被攻击后的主机修复工作

如果你的主机被攻击而被取得操纵权的话，而你也由于了解到主机监控的需要，所以在最短的时间内发现此一事件， 那么该如何针对这个被入侵的主机来修复？那如果你要修复的话，你这个网管人员还需要哪些额外的技能？ 底下我们就来谈一谈。

7.5.1 网管人员应具备的技能

从本章第一小节的分析当中，你会发现网管还真的是挺累的，他需要对操作系统有一定程度的熟悉， 对于程序的运作与权限概念则需要更了解，否则就麻烦了！那除了操作系统的基本概念之外， 咱们网管还需要啥特殊技巧呢？当然需要啊！其实一部主机最常发生问题的状况， 都是由『内部的网络误用所产生的』，所以啊，你只管好主机而已是『没有办法杜绝问题』的啦！ 底下就来谈谈你还需要啥技巧呢？

- 了解什么是需要保护的内容：

  我的天呐，还要知道什么是需要保护的呀？没错，就是如此！由刚刚我们知道的主机入侵方法当中， 不难了解，只要有人坐在你的主机前面，那么任何事都有可能会发生！因此，如果你的主机相当的重要， 请『不要让任何人靠近！』你可以参考一下汤姆克鲁斯在『不可能的任务』里面要窃取一部计算机内的数据的困难度！ ^_^""

  - 硬件：能锁就锁吧！
  - 软件：还包含最重要的数据呢！
- 预防黑客 (Black hats) 的入侵：

  这可不是开玩笑的，什么是黑客呀！这是因为原本在西部电影当中，坏人都是戴黑色帽子的， 所以之前的人们就称网络攻击者为 Black hats 啦！在预防这方面的攻击者时，除了严格管制网络的登入之外， 还需要特别控制原本你的主机中的人物！就我们小网站来说，不要以为好朋友就随便他啦！ 他说要指定密码是跟他的账号相同比较好记，你就答应他！等到人家用他的密码登入你的主机，并破坏你的主机， 那可就得不偿失了！如果是大企业的话，那么员工使用网络时，也要分等级的呢！ ^_^

- 主机环境安全化：

  没什么好讲的，除了多关心，还是多关心！仔细的分析登录档，常常上网看看最新的安全通告，这都是最基础的！ 还包含了以最快的速度更新有问题的软件！因为，越快更新你的软件，就越快可以杜绝黑客的入侵！

- 防火墙规则的订定：

这部份比较麻烦一些啦！因为你必需要不断的测试测试再测试！以取得优化的网络安全设定！ 怎么说呢？要晓得的是，如果你的防火墙规则订定得太多的时候， 那么一个数据封包就要经过越多的关卡才能完整的通过防火墙，以进入到主机内部！嘿嘿！ 这可是相当的花费时间的！会造成主机的效能不彰！特别留意这一点呢！

- 实时维护你的主机：

  就像刚刚说的，你必需要随时维护你的主机，因为，防火墙不是一经设定之后就不用在再他了！ 因为，再严密的防火墙，也会有漏洞的！这些漏洞包括防火规则设定不良、利用较新的侦测入侵技术、 利用你的旧软件的服务漏洞等等！所以，必需要实时维护你的主机呀！这方面除了分析 log files 之外，也可以藉由实时侦测来进行这个工作！例如 PortSentry 就是蛮不错的一套软件呢！

- 良好的教育训练课程：

  不是所有的人都是计算机网络高手，尤其虽然现在信息爆炸但是仍然有很多的机会会遇到计算机白痴呀！ 这个时候，要晓得的是，我们对于内部网域通常没有太多的规范，那如果他用内部的计算机去做坏事怎么办？ 有时候还是无心的～挖哩～所以说，需要特别的教育训练课程呀！这也是公司需要网管的主因之一！

- 完善的备份计划：

  天有不测风云，人有旦夕祸福呀！什么人都不知道什么时候会有大地震、我们也都不知道什么时候会突然的硬盘挂掉去～ 所以说，完善的备份计划是相当重要的！此外，大概没有人会说他的主机是 100% 的安全吧！ 那如果你的系统被入侵，造成数据的损毁时，你要如何复原你的主机啊？呵呵！一个良好的网站管理人员， 无时无刻都会进行重要数据的备份的！很重要啊！这一部份请参考一下基础学习篇之 Linux 主机备份的内容吧！ 我们在后面的远程联机服务器章节内也会提到一个很棒的 rsync 工具，你可以瞧瞧！

7.5.2 主机受攻击后复原工作流程

所谓『百密一疏』啊，人不是神，总会有考虑不周的情况，万一你的主机就因为这『一疏』导致被入侵了， 那该怎么办？由上面的说明当中，我们知道『木马』是很严重的，因为他会在你的系统下开个后门 (Back door) 让攻击者可以登入你的主机，而且还会窜改你 Linux 上面的程序，让你找不到该木马程序！怎么办？

很多朋友都习惯『反正只要将 root 的密码改回来就好了』 这样的观点，事实上，那样一部主机还是有被做为中继站的危险啊！所以， 万一你的主机被入侵了，最好的方法还是『重新安装Linux 』会比较干净！

那该如何重新安装呢？很多朋友一再地安装，却一再地被入侵～为什么呢？因为他没有『记取教训』啊！呵呵！ 底下我们就来谈一谈，一部被入侵的主机应该如何修复比较好？

1. 立即拔除网络线：

   既然发现被入侵了，那么第一件事情就是拿掉网络功能！拿掉网络功能最简单的作法自然就是拔掉网络线了！ 事实上，拿掉网络线最主要的功能除了保护自己之外，还可以保护同网域的其他主机。怎么说呢？举个 2003 年 8 月发病的疾风病毒好了，他会感染同网域之内的其他主机喔！所以，拔除网络线之后，远程的攻击者立即就无法进入你的 Linux 主机，而且你还可以保护网域内的其他相关主机啊！

2. 分析登录文件信息，搜寻可能的入侵途径：

   被入侵之后，决不是只要重新安装就好，还需要额外分析 『为什么我的主机这一次会被入侵，对方是如何入侵的？』， 如果你能够找出问题点，那么不但你的 Linux 功力立刻增强了，主机也会越来越安全喔！ 而如果你不知道如何找出被入侵的可能途径，那么重新安装后，下次还是可能被以同样的方法入侵啊！ 粉麻烦的啦！好了，那该如何找出入侵的途径呢？

   - 分析登录档：低级的 cracker 通常仅是利用工具软件来入侵你的系统，所以我们可以藉由分析一些主要的登录档来找出对方的 IP 以及可能有问题的漏洞。可以分析 /var/log/messages, /var/log/secure 还有利用 last 指令来找出上次登入者的信息。
   - 检查主机开放的服务：很多 Linux 用户常常不晓得自己的系统上面开了多少的服务？我们说过，每个服务都有其漏洞或者是不应该启用的增强型或者是测试型功能， 所以，找出你系统上面的服务，并且检查一下每个服务是否有漏洞，或者是在设定上面有了缺失，然后一个一个的整理吧！
   - 查询 Internet 上面的安全通报： 透过安全通报来了解一下最新的漏洞信息，说不定你的问题就在上面！

3. 重要数据备份：

   主机被入侵后，显得问题相当的严重，为什么呢？因为主机上面有相当重要的数据啊！如果主机上面没有重要的数据， 那么直接重新安装就好了！所以，被入侵之后，检查完了入侵途径，再来就是要备份重要的数据了。 好了，问个问题，什么是『重要数据』？ who, ps, ls 等等指令是重要数据吗？还是 httpd.conf 等配置文件是重要数据？又或者是 /etc/passwd, /etc/shadow 才是重要数据？

   呵呵！基本上，重要的数据应该是『非 Linux 系统上面原有的数据』，例如 /etc/passwd, /etc/shadow, WWW 网页的数据, /home 里面的使用者重要档案等等，至于 /etc/*, /usr/, /var 等目录下的数据，就不见得需要备份了。 注意：不要备份一些 binary 执行文件，因为 Linux 系统安装完毕后本来就有这些档案，此外， 这些档案也很有可能『已经被窜改过了』，那备份这些数据，反而造成下次系统还是不干净！

4. 重新全新安装：

备份完了数据，再来就是重新安装 Linux 系统了。而在这次的安装中， 你最好选择适合你自己的安装软件即可，不要全部软件都给他安装上去啊！挺危险的！

5. 软件的漏洞修补：

 记得啊，重新安装完毕之后，请立即更新你的系统软件，否则还是会被入侵的啦！鸟哥喜欢先在其他比较干净的环境下将 Internet 上面的漏洞修补软件下载下来，然后刻录起来，然后拿到自己的刚刚安装完成的系统上面，mount CD 之后全部给他更新，更新之后，并且设定了相关的防火墙机制，同时进行下一步骤『关闭或移除不需要的服务』后，我才将网络线插上主机的网络卡上！ 因为鸟哥不敢确定在安装完毕后，连上 Internet 去更新软件的这段时间，会不会又受到入侵攻击说....

6. 关闭或移除不需要的服务：

 这个重要性不需要再讲了吧？！启用越少的服务，系统当然可以被入侵的可能性就比较低。

7. 数据回复与恢复服务设定：

 刚刚备份的数据要赶紧的复制回来系统，同时将系统的服务再次的重新开放，请注意，这些服务的设定最好能够再次的确认一下，避免一些不恰当的设定参数在里头喔！

8. 连上 Internet：

 所有的工作都进行的差不多了，那么才将刚刚拿掉的网络线接上来吧！恢复主机的运作了！

经过这一连串的动作后，你的主机应该会恢复到比较干净的环境，此时还不能掉以轻心， 最好还是参考防火墙的设定，并且多方面的参考 Internet 上面一些老手的经验，好让你的主机可以更安全一些！

7.6 重点回顾

- 要管制登入服务器的来源主机，得要了解网络封包的特性，这主要包括 TCP/IP 的封包协议， 以及重要的 Socket Pair ，亦即来源与目标的 IP 与 port 等。在 TCP 封包方面，则还得了解 SYN/ACK 等封包状态；
- 网络封包要进入我们 Linux 本机，至少需要通过 (1)防火墙 (2)服务本身的管理 (3)SELinux (4)取得档案的 rwx 权限等步骤；
- 主机的基本保护之一，就是拥有正确的权限设定。而复杂的权限设定可以利用 ACL 或者是 SELinux 来辅助；
- 关闭 SELinux 可在 /etc/selinux/config 档案内设定，亦可在核心功能中加入 selinux=0 的项目；
- rootkit 为一种取得 root 的工具组，你可以利用 rkhunter 来查询你主机是否被植入 rootkit；
- 网管人员应该注意在员工的教育训练还有主机的完善备份方案上面；
- 一些所谓的黑客软件，几乎都是透过你的 Linux 上面的软件漏洞来攻击 Linux 主机的；
- 软件升级是预防被入侵的最有效方法之一；
- 良好的登录档分析习惯可以在短时间内发现系统的漏洞，并加以修复。

7.7 课后练习

- 我老是发现我的系统怪怪的，似乎有点停顿的模样，怀疑可能是 CPU 负荷太大，所以要去检查一下系统相关的信息。请问，我该以什么指令去检查我的系统相关的信息？可以使用 top, sar, free, ps -aux, uptime, last 等功能去查询系统的相关信息喔！然后再以 kill 之类的指令删除；
- 我怀疑我的系统上面有过多的具有 SUID 的档案存在，导致一般使用者可以随意的取得 root 的权限，请问，我要如何找出这些具有 SUID 权限的档案？因为 SUID 是 4000 这个权限的模样，所以我可以这样做： find / -perm +4000
- 我由国内一些 ftp 网站上下载了 Red Hat 公司释出的软件，我想安装他，但又不知道该软件档案是否被修改过！ 请问我该如何确定这个软件的可用性？利用最简易的 MD5 编码来测试一下，例如『 md5sum 软件名称』，再比对与原始软件释出的 MD5 数据是否相同！？
- 如果我发现使用『 setfacl -m u:dmtsai:rwx /path/to/file 』时，系统却显示『setfacl: Operation not supported』， 你认为是哪里出问题？这是由于你的 filesystem 没有启用 ACL 支持，或者是系统的核心不支持。 请先使用 mount -o remount,acl /mount_point 测试看能否支持 ACL ，若不支持时，则可能是由于核心版本太旧了。
- 如果要设定 dmtsai 可以使用 /home/project 这个目录 (假设 /home 已经支持 ACL)，在该目录内 dmtsai 可以拥有完整的权限。请问该如何设定该目录？除了使用 setfacl -m u:dmtsai:rwx /home/project 之外，还需要设定 setfacl -m m:rwx /home/project ， 因为 ACL 在目录方面，必须透过用户权力及 mask 的逻辑运算后才能生效！
- SELinux 是否为防火墙？SELinux 并非防火墙，他是用来作为更细部权限设定的一个核心模块。
- 良好的密码规划是防备主机的第一要务，请问 Linux 系统当中，关于密码相关的档案与规则设定在哪些档案里面？密码的设定规则在 /etc/login.defs 里面！至于密码档案在 /etc/shadow 内！
- 简易说明，当一部主机被入侵之后，应该如何处理？找出问题、重新安装、漏洞修补、数据还原！请参考本章最后一节的说明。

7.8 参考数据与延伸阅读

- 注1： nmap 的官方网站：http://insecure.org/nmap/
- Fedora 的 SELinux FAQ：http://fedora.redhat.com/docs/selinux-faq/
- SELinux 的发展网站：http://selinux.sourceforge.net/
- Red Hat 之 RHEL 4 的 SELinux 指南：http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/selinux-guide/index.html
- 美国国家安全局的 SELinux 简介：http://www.nsa.gov/selinux/
- Rootkit hunter：http://www.rootkit.nl/projects/rootkit_hunter.html

---

2002/08/12：第一次完成日期！ 2003/08/23：重新编排与增加重点回顾、课后练习 2006/08/31：将旧的文章移动到此处。 2006/09/06：增加 SELinux 的简单说明，增加 ACL 的控件目！ 2010/09/06：将旧的基于 CentOS 4.x 撰写的文章移动至此处。 2010/09/09：因为单纯使用 CentOS ，因此取消了 apt 的更新功能啰！ 2010/09/21：将旧的 限制联机埠口 与 网络升级软件 挪动到新的目录去了！ 2010/09/21：还有许多数据报括(1)重点回顾/(2)课后练习/(3)延伸阅读的部分都还没有汇整好。只是先公告这一版而已。 2011/07/20：将基于 CentOS 5.x 的文章移动到此处。 2011/07/21：SELinux 还真是怪异啊！修改成 CentOS 6.x 的模样啦！

第八章、路由观念与路由器设定

最近更新日期：2011/07/22

如果说 IP 是门牌，那么邮差如何走到你家就是『路由』的功能啦！局域网络如果想成是条巷子，那么路由器就是那间巷子内的邮局！ 其实本章应该是第二章网络基础的延伸，将网络的设定延伸到整个区网的路由器上而已。那何时会用到路由器？ 如果你的环境中需要将整批 IP 再区隔出不同的广播区段时，那么就得要透过路由器的封包转递能力了。 本章是下一章防火墙与 NAT 的基础，得先看完才比较容易理解下一章想要讨论的事情喔！

8.1 路由

我们在第二章网络基础里面谈到过路由的相关概念， 他最大的功能就是在帮我们规划网络封包的传递方式与方向。至于路由的观察则可以使用 route 这个指令来查阅与设定。 好了，那么路由的形式有哪些？你又该如何确认路由是否正确呢？

8.1.1 路由表产生的类型

如同第二章网络基础里面谈到的，每一部主机都有自己的路由表， 也就是说，你必须要透过你自己的路由表来传递你主机的封包到下一个路由器上头。 若传送出去后，该封包就得要透过下一个路由器的路由表来传送了，此时与你自己主机的路由表就没有关系啦！ 所以说，如果网络上面的某一部路由器设定错误，那...封包的流向就会发生很大的问题。 我们就得要透过 traceroute 来尝试了解一下每个 router 的封包流向啰。

OK！那你自己主机的路由表到底有哪些部分呢？我们以底下这个路由表来说明：

```
[root@www ~]# route -n
Kernel IP routing table
Destination   Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0 &lt;== 1
169.254.0.0   0.0.0.0         255.255.0.0     U     1002   0        0 eth0 &lt;== 2
0.0.0.0       192.168.1.254   0.0.0.0         UG    0      0        0 eth0 &lt;== 3
```

首先，我们得知道在 Linux 系统下的路由表是由小网域排列到大网域， 例如上面的路由表当中，路由是由『 192.168.1.0/24 --> 169.254.0.0/16 --> 0.0.0.0/0 (预设路由)』来排列的。 而当主机的网络封包需要传送时，就会查阅上述的三个路由规则来了解如何将该封包传送出去。 那你会不会觉得奇怪，为什么会有这几个路由呢？其实路由表主要有这几种情况来设计的：

- 依据网络接口产生的 IP 而存在的路由：

  例如 192.168.1.0/24 这个路由的存在是由于鸟哥的这部主机上面拥有 192.168.1.100 这个 IP 的关系！ 也就是说，你主机上面有几个网络接口的存在时，该网络接口就会存在一个路由才对。 所以说，万一你的主机有两个网络接口时，例如 192.168.1.100, 192.168.2.100 时，那路由至少就会有：

  ```
  [root@www ~]# ifconfig eth1 192.168.2.100
  [root@www ~]# route -n
  Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
  192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
  192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
  169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
  0.0.0.0         192.168.1.254   0.0.0.0         UG    0      0        0 eth0
  ```

- 手动或预设路由(default route)：

  你可以使用 route 这个指令手动的给予额外的路由设定，例如那个预设路由 (0.0.0.0/0) 就是额外的路由。 使用 route 这个指令时，最重要的一个概念是：『你所规划的路由必须要是你的装置 (如 eth0) 或 IP 可以直接沟通 (broadcast) 的情况』才行。举例来说，以上述的环境来看， 我的环境里面仅有 192.168.1.100 及 192.168.2.100 ，那我如果想要连接到 192.168.5.254 这个路由器时， 下达：

  ```
  [root@www ~]# route add -net 192.168.5.0  \
  &gt; netmask 255.255.255.0 gw 192.168.5.254
  SIOCADDRT: No such process
  ```

  看吧！系统就会响应没有办法连接到该网域，因为我们的网络接口与 192.168.5.0/24 根本就没有关系嘛！ 那如果 192.168.5.254 真的是在我们的实体网络连接上，并且与我们的 eth0 连接在一起，那其实你应该是这样做：

  ```
  [root@www ~]# route add -net 192.168.5.0  \
  &gt; netmask 255.255.255.0 dev eth0
  [root@www ~]# route -n
  Kernel IP routing table
  Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
  192.168.5.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
  192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
  192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
  169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
  0.0.0.0         192.168.1.254   0.0.0.0         UG    0      0        0 eth0
  ```

  这样你的主机就会直接用 eth0 这个装置去尝试连接 192.168.5.254 了！ 另外，上面路由输出的重点其实是那个『Flags 的 G 』了！因为那个 G 代表使用外部的装置作为 Gateway 的意思！而那个 Gateway (192.168.1.254) 必须要在我们的已存在的路由环境中。 这可是很重要的概念喔！ ^_^

- 动态路由的学习：

  除了上面这两种可以直接使用指令的方法来增加路由规则之外，还有一种透过路由器与路由器之间的协商以达成动态路由的环境， 不过，那就需要额外的软件支持了，例如：zebra (http://www.zebra.org/) 或 CentOS 上面的 Quagga (http://www.quagga.net/) 这几个软件了！

事实上，在 Linux 的路由规则都是透过核心来达成的，所以这些路由表的规则都是在核心功能内啊！也就是在内存当中喔！ ^_^

8.1.2 一个网卡绑多个 IP： IP Alias 的测试用途

我们在第五章的 ifconfig 指令里面谈过 eth0:0 这个装置吧？这个装置可以在原本的 eth0 上面模拟出一个虚拟接口出来，以让我们原本的网络卡具有多个 IP ，具有多个 IP 的功能就被称为 IP Alias 了。而这个 eth0:0 的装置可以透过 ifconfig 或 ip 这两个指令来达成， 关于这两个指令的用途请翻回去之前的章节阅读，这里不再浪费篇幅啊！

那你或许会问啊：『这个 IP Alias 有啥用途啊？』好问题！这个 IP Alias 最大的用途就是可以让你用来『应急』！ 怎么说呢？我们就来聊一聊他的几个常见的用途好了：

- 测试用：

  怎么说用来测试呢？举例来说，现在使用 IP 分享器的朋友很多吧，而 IP 分享器的设定通常是使用 WWW 接口来提供的。这个 IP 分享器通常会给予一个私有 IP 亦即是 192.168.0.1 来让用户开启 WWW 接口的浏览。问题来了，那你要如何连接上这部 IP 分享器呢？嘿嘿！在不更动既有的网络环境下，你可以直接利用：

  ```
  [root@www ~]# ifconfig [device] [ IP ] netmask [netmask ip] [up&#124;down]
  [root@www ~]# ifconfig eth0:0 192.168.0.100 netmask 255.255.255.0 up
  ```

  来建立一个虚拟的网络接口，这样就可以立刻连接上 IP 分享器了，也不会更动到你原本的网络参数设定值哩！

- 在一个实体网域中含有多个 IP 网域：

  另外，如果像是在补习班或者是学校单位的话，由于原本的主机网络设定最好不要随便修改， 那如果要让同学们大家互通所有的计算机信息时，就可以让每个同学都透过 IP Alias 来设定同一网域的 IP ， 如此大家就可以在同一个网段内进行各项网络服务的测试了，很不错吧！

- 既有设备无法提供更多实体网卡时：

  如果你的这部主机需要连接多个网域，但该设备却无法提供安装更多的网卡时，你只好勉为其难的使用 IP Alias 来提供不同网段的联机服务了！

不过，你需要知道的是：所有的 IP Alias 都是由实体网卡仿真来的，所以当要启动 eth0:0 时，eth0 必须要先被启动才行。而当 eth0 被关闭后，所以 eth0:n 的模拟网卡将同时也被关闭。这得先要了解才行， 否则常常会搞错启动的装置啊！在路由规则的设定当中，常常需要进行一些测试，那这个 IP Alias 就派的上用场了。 尤其是学校单位的练习环境当中！

基本上，除非有特殊需求，否则建议你要有多个 IP 时，最好在不同的网卡上面达成，如果你真的要使用 IP Alias 时，那么如何在开机的时候就启动 IP alias 呢？方法有很多啦！包括将上面用 ifconfig 启动的指令写入 /etc/rc.d/rc.local 档案中 (但使用 /etc/init.d/network restart 时，该 IP alias 无法被重新启动)， 但鸟哥个人比较建议使用如下的方式来处理：

- 透过建立 /etc/sysconfig/network-scripts/ifcfg-eth0:0 配置文件

  举例来说，你可以透过底下这个方法来建立一个虚拟设备的配置文件案：

```
[root@www ~]# cd /etc/sysconfig/network-scripts
[root@www network-scripts]# vim ifcfg-eth0:0
DEVICE=eth0:0            &lt;==相当重要！一定要与文件名相同的装置代号！
ONBOOT=yes
BOOTPROTO=static
IPADDR=192.168.0.100
NETMASK=255.255.255.0

[root@www network-scripts]# ifup eth0:0
[root@www network-scripts]# ifdown eth0:0
[root@www network-scripts]# /etc/init.d/network restart
```

关于装置的配置文件案内的更多参数说明， 请参考第四章 4.2.1 手动设定 IP 参数的相关说明， 在此不再叙述！使用这个方法有个好处，就是当你使用『 /etc/init.d/network restart 』时，系统依旧会使用你的 ifcfg-eth0:0 档案内的设定值来启动你的虚拟网卡喔！另外，不论 ifcfg-eth0:0 内的 ONBOOT 设定值为何，只要 ifcfg-eth0 这个实体网卡的配置文件中， ONBOOT 为 yes 时，开机就会将全部的 eth0:n 都启动。

透过这个简单的方法，你就可以在开机的时候启动你的虚拟接口而取得多个 IP 在同一张网卡上了。不过需要注意的是， 如果你的这张网卡分别透过 DHCP 以及手动的方式来设定你的 IP 参数，那么 dhcp 的取得务必使用实体网卡，亦即是 eth0 之类的网卡代号，而手动的就以 eth0:0 之类的代号来设定较佳。

Tips: 在旧版的 CentOS 4.x 中，如果你的 eth0 是使用 DHCP 来取得 IP 参数的话，那么由于 ifup 及 /etc/init.d/network 这两个 script 内程序代码撰写的方式，将会导致 ifcfg-eth0:0 这个配置文件不会被使用到喔！不过这个问题在 CentOS 5.x 以后的版本中已经被克服啰！



8.1.3 重复路由的问题

很多朋友可能都有一个可爱的想法，那就是：『我可不可以利用两张网卡， 利用两个相同网域的 IP 来增加我这部主机的网络流量』？事实上这是一个可行的方案， 不过必须要透过许多的设定来达成，若你有需求的话，可以参考网中人大哥写的这一篇 (注1)：

- 带宽负载平衡 (http://www.study-area.org/tips/multipath.htm)

如果只是单纯的以为设定好两张网卡的 IP 在同一个网域就能够增加你主机的两倍流量，那可就大错特错了～ 为什么呢？还记得我们在路由表规则里面提过网络封包的传递主要是依据主机内的路由表规则吧！ 那如果你有两张网络卡时，假设：(底下信息请思考，不用实作！)

- eth0 : 192.168.0.100
- eth1 : 192.168.0.200

那你的路由规则会是如何呢？理论上会变成这样：

```
[root@www ~]# route -n
Kernel IP routing table
Destination     Gateway   Genmask         Flags Metric Ref   Use Iface
192.168.0.0     0.0.0.0   255.255.255.0   U     0      0       0 eth1
192.168.0.0     0.0.0.0   255.255.255.0   U     0      0       0 eth0
```

也就是说，(1)当要主动发送封包到 192.168.0.0/24 的网域时，都只会透过第一条规则 ，也就是透过 eth1 来传出去！ (2)在响应封包方面，不管是由 eth0 还是由 eth1 进来的网络封包，都会透过 eth1 来回传！这可能会造成一些问题，尤其是一些防火墙的规则方面，很可能会发生一些严重的错误， 如此一来，根本没有办法达成负载平衡，也不会有增加网络流量的效果！ 更惨的是，还可能发生封包传递错误的情况呐！所以说，同一部主机上面设定相同网域的 IP 时， 得要特别留意你的路由规则，一般来说，不应该设定同一的网段的不同 IP 在同一部主机上面。 例如上面的案例就是一个不好的示范啊！

Tips: 为什么会特别强调这个观念呢？大约 2000 年前后，鸟哥刚接触 Linux 时，由于当时的网络速度相当缓慢， 为了提升网络流量鸟哥费尽心思啊～后来想到说，如果有两片网卡，不就可以增加流量了吗？于是就设定了两个同网域的 IP 在一部主机的两张网卡上，结果呢？很多服务都无法连通了！就是因为痛过，所以才有更强烈的印象啊！错误经验学习法则 ^_^！



8.2 路由器架设

我们知道在局域网络里面的主机可以透过广播的方式来进行网络封包的传送，但在不同网段内的主机想要互相联机时，就得要透过路由器了。 那么什么是路由器？他的主要功能是什么？底下我们就来聊一聊！

8.2.1 什么是路由器与 IP 分享器

既然主机想要将数据传送到不同的网域时得透过路由器的帮忙，所以啦，路由器的主要功能就是：『转递网络封包』啰！也就是说，路由器会分析来源端封包的 IP 表头，在表头内找出要送达的目标 IP 后，透过路由器本身的路由表 (routing table) 来将这个封包向下一个目标 (next hop) 传送。这就是路由器的功能。 那么路由器的功能可以如何达成呢？目前有两种方法可以达成：

- 硬件功能：例如 Cisco, TP-Link, D-Link (注2) 等公司都有生产硬件路由器， 这些路由器内有嵌入式的操作系统，可以负责不同网域间的封包转译与转递等功能；
- 软件功能：例如 Linux 这个操作系统的核心就有提供封包转递的能力。

高阶的路由器可以连结不同的硬设备，并且可以转译很多不同的封包格式，通常...价格也不便宜啊！ 在这个章节里面，我们并没有要探讨这么高阶的咚咚，仅讨论在以太网络里头最简单的路由器功能： 连接两个不同的网域。嘿嘿！这个功能 Linux 个人计算机就可以达成了！那怎么达成呢？

- 打开核心的封包转递 (IP forward) 功能

就如同路由表是由 Linux 的核心功能所提供的，这个转递封包的能力也是 Linux 核心所提供，那如何观察核心是否已经有启动封包转递呢？很简单啊，观察核心功能的显示档案即可，如下所示：

```
[root@www ~]# cat /proc/sys/net/ipv4/ip_forward
0  &lt;== 0 代表没有启动， 1 代表启动了
```

要让该档案的内容变成启动值 1 最简单的方是就是使用：『echo 1 > /proc/sys/net/ipv4/ip_forward』即可。 不过，这个设定结果在下次重新启动后就会失效。因此，鸟哥建议您直接修改系统配置文件的内容，那就是 /etc/sysctl.conf 来达成开机启动封包转递的功能喔。

```
[root@www ~]# vim /etc/sysctl.conf
# 将底下这个设定值修改正确即可！ (本来值为 0 ，将它改为 1 即可)
net.ipv4.ip_forward = 1

[root@www ~]# sysctl -p  &lt;==立刻让该设定生效
```

sysctl 这个指令是在核心工作时用来直接修改核心参数的一个指令，更多的功能可以参考 man sysctl 查询。 不要怀疑！只要这个动作，你的 Linux 就具有最简单的路由器功能了。而由于 Linux 路由器的路由表设定方法的不同，通常路由器规划其路由的方式就有两种：

- 静态路由：直接以类似 route 这个指令来直接设定路由表到核心功能当中，设定值只要与网域环境相符即可。 不过，当你的网域有变化时，路由器就得要重新设定；
- 动态路由：透过类似 Quagga 或 zebra 软件的功能，这些软件可以安装在 Linux 路由器上， 而这些软件可以动态的侦测网域的变化，并直接修改 Linux 核心的路由表信息， 你无须手动以 route 来修改你的路由表信息喔！

了解了路由器之后，接下来你可能需要了解到什么是 NAT (Network Address Translation, 网络地址转换) 服务器， NAT 是啥？其实 IP 分享器就是最简单的 NAT 服务器啦！嘿嘿，了解了吗？没错， NAT 可以达成 IP 分享的功能， 而 NAT 本身就是一个路由器，只是 NAT 比路由器多了一个『 IP 转换』的功能。怎么说呢？

- 一般来说，路由器会有两个网络接口，透过路由器本身的 IP 转递功能让两个网域可以互相沟通网络封包。 那如果两个接口一边是公共 IP (public IP) 但一边是私有 IP (private IP) 呢？ 由于私有 IP 不能直接与公共 IP 沟通其路由信息，此时就得要额外的『 IP 转译』功能了；
- Linux 的 NAT 服务器可以透过修改封包的 IP 表头数据之来源或目标 IP ，让来自私有 IP 的封包可以转成 NAT 服务器的公共 IP ，就可以连上 Internet ！

所以说，当路由器两端的网域分别是 Public 与 Private IP 时，才需要 NAT 的功能！ NAT 功能我们会在下一章防火墙时谈及， 这个章节仅谈论一下路由器而已啊！ ^_^

8.2.2 何时需要路由器

一般来说，计算机数量小于数十部的小型企业是无须路由器的，只需要利用 hub/switch 串接各部计算机， 然后透过单一线路连接到 Internet 上即可。不过，如果是超过数百部计算机的大型企业环境， 由于他们的环境通常需要考虑如下的状况，因此才需要路由器的架设：

- 实体线路之布线及效能的考虑：

  在一栋大楼的不同楼层要串接所有的计算机可能有点难度，那可以透过每个楼层架设一部路由器， 并将每个楼层路由器相连接，就能够简单的管理各楼层的网络； 此外，如果各楼层不想架设路由器，而是直接以网络线串接各楼层的 hub/switch 时， 那由于同一网

域的数据是透过广播来传递的，那当整个大楼的某一部计算机在广播时， 所有的计算机将会予以回应，哇！会造成大楼内网络效能的问题；所以架设路由器将实体线路分隔，就有助于这方面的网络效能；

- 部门独立与保护数据的考虑：

  在阅读过第二章网络基础后，你就会晓得， 只要实体线路是连接在一起的，那么当数据透过广播时，你就可以透过类似 tcpdump 的指令来监听封包数据， 并且予以窃取～所以，如果你的部门之间的数据可能需要独立， 或者是某些重要的资料必须要在公司内部也予以保护时，可以将那些重要的计算机放到一个独立的实体网域， 并额外加设防火墙、路由器等连接上公司内部的网域。

路由器就只是一个设备，要如何使用端看你的网络环境的规划！上面仅是举出一些应用案例。 底下我们先就架设一个静态路由的路由器来玩一玩吧！

8.2.3 静态路由之路由器

假设在贵公司的网络环境当中，除了一般职员的工作用计算机是直接连接到对外的路由器来连结因特网， 在内部其实还有一个部门需要较安全的独立环境，因此这部份的网络规划可能是这样的情况 (参考图 3.2-1 内容延伸而来)：

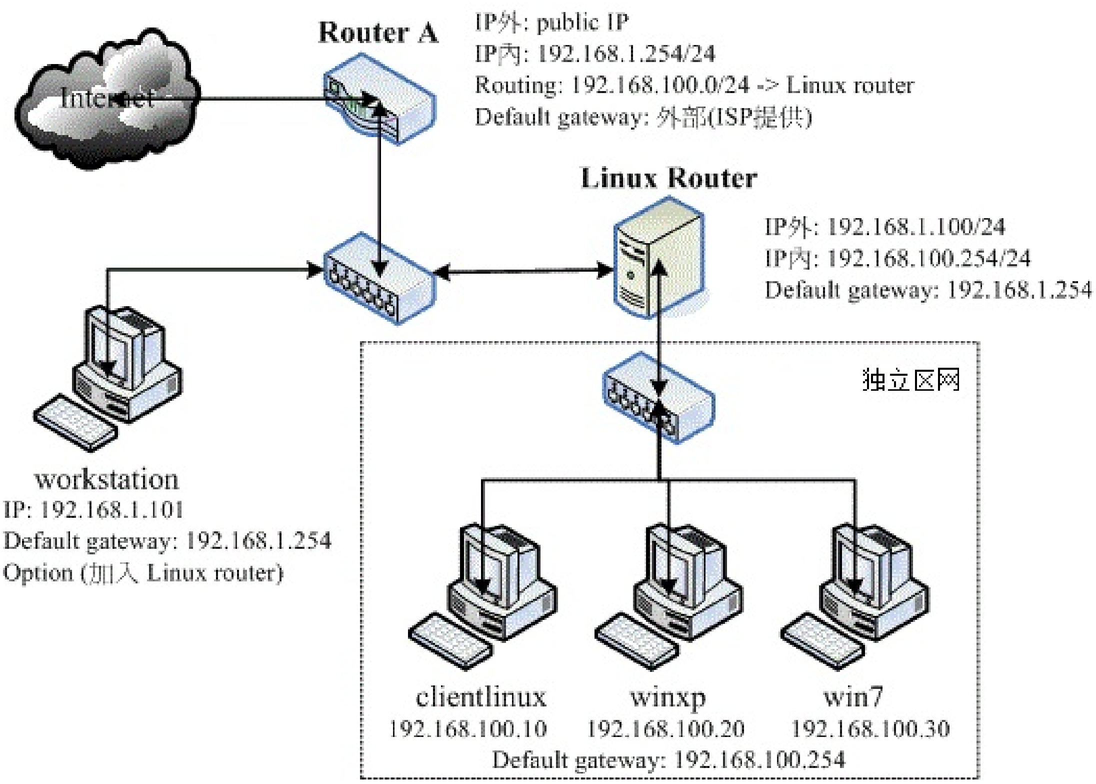


图 8.2-1、静态路由之路由器架构示意图

以上图的架构来说，这家公司主要有两个 class C 的网段，分别是：

- 一般区网(192.168.1.0/24) ：包括 Router A, workstation 以及 Linux Router 三部主机所构

成；
- 保护内网(192.168.100.0/24)：包括 Linux Router, clientlinux, winxp, win7 等主机所构成。

其中 192.168.1.0/24 是用来做为一般员工连接因特网用的，至于 192.168.100.0/24 则是给特殊的部门用的。workstation 代表的是一般员工的计算机，clientlinux 及 winxp, win7 则是特殊部门的工作用计算机， Linux Router 则是这个特殊部门用来连接到公司内部网域的路由器。在这样的架构下， 该特殊部门的封包就能够与公司其他部门作实体的分隔了。

由上图你也不难发现，只要是具有路由器功能的设备 (Router A, Linux Router) 都会具有两个以上的接口， 分别用来沟通不同的网域，同时该路由器也都会具有一个预设路由啊！ ^_^！另外，你还可以加上一些防火墙的软件在 Linux Router 上，以保护 clientlinux, winxp, win7 呢！

那我们先来探讨一下联机的机制好了，先从 clientlinux 这部计算机谈起。如果 clientlinux 想要连上 Internet，那么他的联机情况会是如何？

- 发起联机需求：clientlinux --> Linux Router --> Router A --> Internet
- 响应联机需求：Internet --> Router A --> Linux Router --> clientlinux

观察一下两部 Router 的设定，要达到上述功能，则 Router A 必须要有两个接口，一个是对外的 Public IP 一个则是对内的 Private IP ，因为 IP 的类别不同，因此 Router A 还需要额外增加 NAT 这个机制才行，这个机制我们在后续章节会继续谈到。 除此之外，Router A 并不需要什么额外的设定。至于 Linux Router 就更简单了！什么事都不用作，将两个网络适配器设定两个 IP ， 并且启动核心的封包转递功能，立刻就架设完毕了！非常简单！我们就来谈一谈这几个机器的设定吧！

- Linux Router

在这部主机内需要有两张网卡，鸟哥在这里将他定义为 (假设你已经将刚刚实作的 eth0:0 取消掉了)：

- eth0: 192.168.1.100/24
- eth1: 192.168.100.254/24

```
# 1\. 再看看 eth0 的设定吧！虽然我们已经在第四章就搞定了：
[root@www ~]# vim /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE="eth0"
HWADDR="08:00:27:71:85:BD"
NM_CONTROLLED="no"
ONBOOT="yes"
BOOTPROTO=none
IPADDR=192.168.1.100
NETMASK=255.255.255.0
GATEWAY=192.168.1.254   &lt;==最重要的设定啊！透过这部主机连出去的！

# 2\. 再处理 eth1 这张之前一直都没有驱动的网络卡吧！
[root@www ~]# vim /etc/sysconfig/network-scripts/ifcfg-eth1
DEVICE="eth1"
HWADDR="08:00:27:2A:30:14"
NM_CONTROLLED="no"
ONBOOT="yes"
BOOTPROTO="none"
IPADDR=192.168.100.254
NETMASK=255.255.255.0

# 3\. 启动 IP 转递，真的来实作成功才行！
[root@www ~]# vim /etc/sysctl.conf
net.ipv4.ip_forward = 1
# 找到上述的设定值，将默认值 0 改为上述的 1 即可！储存后离开去！
[root@www ~]# sysctl -p
[root@www ~]# cat /proc/sys/net/ipv4/ip_forward
1   &lt;==这就是重点！要是 1 才可以哟！

# 4\. 重新启动网络，并且观察路由与 ping Router A
[root@www ~]# /etc/init.d/network restart
[root@www ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.100.0   0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.1.254   0.0.0.0         UG    0      0        0 eth0
# 上面的重点在于最后面那个路由器的设定是否正确哟！

[root@www ~]# ping -c 2 192.168.1.254
PING 192.168.1.254 (192.168.1.254) 56(84) bytes of data.
64 bytes from 192.168.1.254: icmp_seq=1 ttl=64 time=0.294 ms
64 bytes from 192.168.1.254: icmp_seq=2 ttl=64 time=0.119 ms &lt;==有回应即可

# 5\. 暂时关闭防火墙！这一步也很重要喔！
[root@www ~]# /etc/init.d/iptables stop
```

有够简单吧！而且透过最后的 ping 我们也知道 Linux Router 可以连上 Router A 啰！这样你的 Linux Router 就 OK 了呐！此外，CentOS 6.x 默认的防火墙规则会将来自不同网卡的沟通封包剔除，所以还得要暂时关闭防火墙才行。 接下来则是要设定 clientlinux 这个被保护的内部主机网络啰。

- 受保护的网域，以 clientlinux 为例

不论你的 clientlinux 是哪一种操作系统，你的环境都应该是这样的 (图 8.2-1)：

- IP: 192.168.100.10
- netmask: 255.255.255.0
- gateway: 192.168.100.254

- hostname: clientlinux.centos.vbird
- DNS: 168.95.1.1

以 Linux 操作系统为例，并且 clientlinux 仅有 eth0 一张网卡时，他的设定是这样的：

```
[root@clientlinux ~]# vim /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE="eth0"
NM_CONTROLLED="no"
ONBOOT="yes"
BOOTPROTO=none
IPADDR=192.168.100.10
NETMASK=255.255.255.0
GATEWAY=192.168.100.254  &lt;==这个设定最重要啦！
DNS1=168.95.1.1          &lt;==有这个就不用自己改 /etc/resolv.conf

[root@clientlinux ~]# /etc/init.d/network restart
[root@clientlinux ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.100.0   0.0.0.0         255.255.255.0   U     1      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
0.0.0.0         192.168.100.254 0.0.0.0         UG    0      0        0 eth0

[root@clientlinux ~]# ping -c 2 192.168.100.254 &lt;==ping自己的gateway(会成功)
[root@clientlinux ~]# ping -c 2 192.168.1.254   &lt;==ping外部的gateway(会失败)
```

最后一个动作有问题哟！怎么会连 ping 都没有办法 ping 到 Router A 的 IP 呢？如果连 ping 都没有办法给予回应的话， 那么表示我们的联机是有问题的！再从刚刚的响应联机需求流程来看一下吧！

- 发起联机：clientlinux --> Linux Router (OK) --> Router A (OK)
- 回应联机：Router A (此时 router A 要响应的目标是 192.168.100.10)，Router A 仅有 public 与 192.168.1.0/24 的路由，所以该封包会由 public 界面再传出去，因此封包就回不来了...

发现了吗？网络是双向的，此时封包出的去，但是非常可怜的，封包回不来～那怎办呢？只好告知 Router A 当路由规则碰到 192.168.100.0/24 时，要将该封包传 192.168.1.100 就是了！所以你要这样进行。

- 特别的路由规则： Router A 所需路由

假设我的 Router A 对外的网卡为 eth1 ，而内部的 192.168.1.254 则是设定在 eth0 上头。 那怎么在 Router A 增加一条路由规则呢？很简单啊！直接使用 route add 去增加即可！如下所示的情况：

```
[root@routera ~]# route add -net 192.168.100.0 netmask 255.255.255.0 \
&gt;  gw 192.168.1.100
```

不过这个规则并不会写入到配置文件，因此下次重新启动这个规则就不见了！所以，你应该要建立一个路由配置文件。 由于这个路由是依附在 eth0 网卡上的，所以配置文件的档名应该要是 route-eth0 喔！这个配置文件的内容当中，我们要设定 192.168.100.0/24 这个网域的 gateway 是 192.168.1.100，且是透过 eth0 ，那么写法就会变成：

```
[root@routera ~]# vim /etc/sysconfig/network-scripts/route-eth0
192.168.100.0/24 via 192.168.1.100 dev eth0
目标网域             透过的gateway      装置

[root@routera ~]# route -n
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
120.114.142.0   0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.100.0   192.168.1.100   255.255.255.0   UG    0      0        0 eth0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         120.114.142.254 0.0.0.0         UG    0      0        0 eth1
```

上述观察的重点在于有没有出现 192.168.100.0 那行路由！如果有的话，请 ping 192.168.100.10 看看能不能有回应？ 然后再到 clientlinux 上面去 ping 192.168.1.254 看看有没有响应，你就知道设定成功啰！好了，既然内部保护网络已经可以连上 Internet 了，那么是否代表 clientlinux 可以直接与一般员工的网域，例如 workstation 进行联机呢？我们依旧透过路由规则来探讨一下，当 clientlinux 要直接联机到 workstation 时，他的联机方向是这样的(参考图 8.2-1)：

- 联机发起： clientlinux --> Linux Router (OK) --> workstation (OK)
- 回应联机： workstation (联机目标为 192.168.100.10，因为并没有该路由规则，因此联机丢给 default gateway，亦即是 Router A) --> Router A (OK) --> Linux Router (OK) --> clientlinux

有没有发现一个很可爱的传输流程？联机发起是没有问题啦，不过呢，响应联机竟然会偷偷透过 Router A 来帮忙呦！ 这是因为 workstation 与当初的 Router A 一样，并不知道 192.168.100.0/24 在 192.168.1.100 里面啦！不过，反正 Router A 已经知道了该网域在 Linux Router 内，所以，该封包还是可以顺利的回到 clientlinux 就是了。

- 让 workstation 与 clientlinux 不透过 Router A 的沟通方式

如果你不想要让 workstation 得要透过 Router A 才能够联机到 clientlinux 的话，那么就得要与 Router A 相同，增加那一条路由规则啰！如果是 Linux 的系统，那么如同 Router A 一样的设定如下：

```
[root@workstation ~]# vim /etc/sysconfig/network-scripts/route-eth0
192.168.100.0/24 via 192.168.1.100 dev eth0

[root@workstation ~]# /etc/init.d/network restart
[root@www ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.100.0   192.168.1.100   255.255.255.0   UG    0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth0
0.0.0.0         192.168.1.254   0.0.0.0         UG    0      0        0 eth0
```

最后只要 clientlinux 使用 ping 可以连到 workstation，同样的，workstation 也可以 ping 到 clientlinux 的话，就表示你的设定是 OK 的啦！搞定！而透过这样的设定方式，你也可以发现到一件事，那就是：『路由是双向的，你必须要了解出去的路由与回来时的规则』。 举例来说，在预设的情况下 (Router A 与 workstation 都没有额外的路由设定时)，其实封包是可以由 clientlinux 联机到 workstation 的，但是 workstation 却没有相关的路由可以响应到 clientlinux ～所以上头才会要你在 Router A 或者是 workstation 上面设定额外的路由规则啊！这样说，瞭了吧？ ^_^

用 Linux 作一个静态路由的 Router 很简单吧！以上面的案例来说，你在 Linux Router 上面几乎没有作什么额外的工作，只要将网络 IP 与网络接口对应好启动，然后加上 IP Forward 的功能， 让你的 Linux 核心支持封包转递，然后其他的工作咱们的 Linux kernel 就主动帮你搞定了！真是好简单！

不过这里必须要提醒的是，如果你的 Linux Router 有设定防火墙的话， 而且还有设定类似 NAT 主机的 IP 伪装技术，那可得特别留意，因为还可能会造成路由误判的问题～ 上述的 Linux Router 当中『并没有使用到任何 NAT 的功能』喔！特别给他留意到！

8.3 动态路由器架设：quagga (zebra + ripd)

在一般的静态路由器上面，我们可以透过修改路由配置文件 (route-ethN) 来设定好既定的路由规则，让你的路由器运作顺利。不过， 这样的方法总是觉得很讨厌！如果某天因为组织的再造导致需要重新规划子网网段，如此一来，你就得要在图 8.2-1 的 Router A 与 Linux Router 再次的处理与检查路由规则，真是有够麻烦的～ 那能不能让路由器自己学习新的路由，来达成自动增加该笔路由的信息呢？

上述的功能就是所谓的动态路由。动态路由通常是用在路由器与路由器之间的沟通，所以要让你的路由器具有动态路由的功能， 你必须要了解到对方路由器上面所提供的动态路由协议才行，这样两部路由器才能够透过该协议来沟通彼此的路由规则。 目前常见的动态路由协议有：RIPv1, RIPv2, OSPF, BGP 等。

想要在 CentOS 上面搞定这些动态路由的相关机制，那就得要使用 quagga 这个软件啦！这个软件是 zebra 计划的延伸， 相关的官网说明可以参考文后的参考数据(注3)。既然要玩 quagga ，自然就得要先安装他啰！赶紧处理吧！

```
[root@www ~]# yum install quagga
[root@www ~]# ls -l /etc/quagga
-rw-r--r--. 1 root   root      406 Jun 25 20:19 ripd.conf.sample
-rw-r-----. 1 quagga quagga     26 Jul 22 11:11 zebra.conf
-rw-r--r--. 1 root   root      369 Jun 25 20:19 zebra.conf.sample
.....(其他省略).....
```

这个软件所提供的各项动态路由协议都放置到 /etc/quagga/ 目录内，底下我们以较为简单的 RIPv2 协议来处理动态路由， 不过你得要注意的是，不论你要启动什么动态路由协议，那个 zebra 都必须要先启动才行！这是因为：

- zebra 这个 daemon 的功能在更新核心的路由规则；
- RIP 这个 daemon 则是在向附近的其他 Router 沟通协调路由规则的传送与否。

而各个路由服务的配置文件都必须要以 /etc/quagga/*.conf 的档名来储存才行，如上表我们可以发现 zebra 这个服务是有设定好了，不过 ripd 的档名却不是 .conf 结尾。所以我们必须要额外作些设定才行。

为了练习一下我们的 quagga ，当然得要设计一下可能的网络联机啰～假设网络联机的图标如下，共有三个区网的网段， 其中最大的是 192.168.1.0/24 这个外部区网，另有两个内部区网分别是 192.168.100.0/24 及 192.168.200.0/24 。

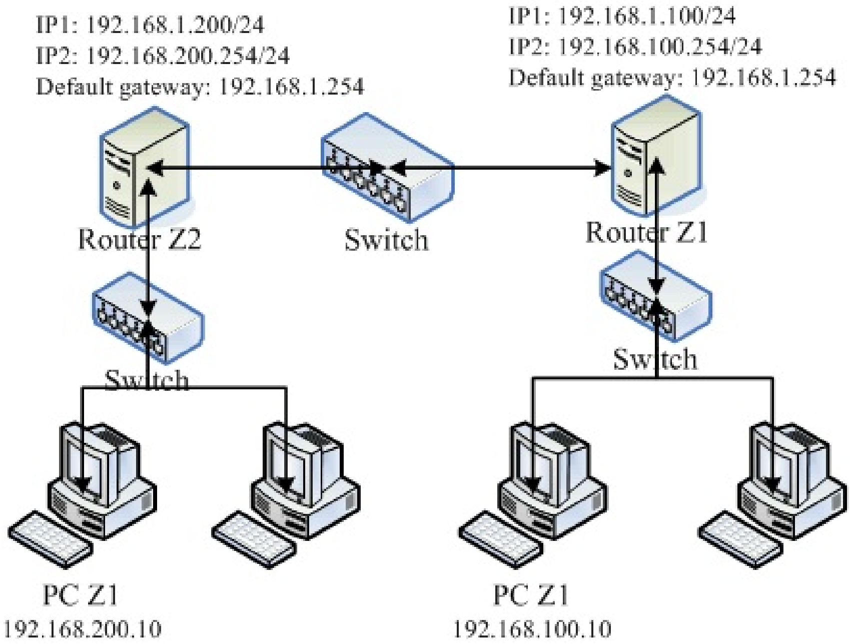


图 8.3-1、练习动态路由所设定的网络联机示意图

上图的两部 Linux Router 分别负责不同的网域，其中 Router Z1 是上个小节设定好之后就保留的，左边的 Router Z2 则是需要额外设定的路由器喔！两部 Router 可以透过 192.168.1.0/24 这个网域来沟通。在没有设定额外路由规则的情况下，那个 PC Z1 与 PC Z2 是无法沟通的！另外，quagga 必须要同时安装在两部 Linux Router 上头才行， 而且我们只要设定好这两部主机的网络接口 (eth0, eth1) 后，不需要手动输入额外的路由设定喔！可以透过 RIP 这个路由协议来搞定的！

- 1. 将所有主机的 IP 设定妥当：

  这是最重要的吧？请将这四部主机 (Router Z1, Router Z2, PC Z1, PC Z2) 的网络参数，按照图 8.3-1 的模样设置妥当。 设置的方式请参考本章上一小节，或者是依据第四章的 4.2.1 来设定啰，这里不再重复说明了。 另外，在 Router Z1, Z2 的部分还得要加上修改 ip*forwrad* 参数！亦即是 */etc/sysctl.conf* 的设定值喔！ 这个鸟哥也常常忘记 @@。

- 2. 在两部 Router 上面设定 zebra ：

  我们先设定图 8.3-1 右手边那一部 Router Z1，关于 zebra.conf 你可以这样设定的：

```
# 1\. 先设定会影响动态路由服务的 zebra 并且启动 zebra
[root@www ~]# vim /etc/quagga/zebra.conf
hostname www.centos.vbird           &lt;==给予这个路由器一个主机名，随便取！
password linuxz1                    &lt;==给予一个密码！
enable password linuxz1             &lt;==将这个密码生效！
log file /var/log/quagga/zebra.log  &lt;==将所有 zebra 产生的信息存到登录文件中

[root@www ~]# /etc/init.d/zebra start
[root@www ~]# chkconfig zebra on
[root@www ~]# netstat -tunlp &#124; grep zebra
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address   Foreign Address   State    PID/Program name
tcp        0      0 127.0.0.1:2601  0.0.0.0:*         LISTEN   4409/zebra
```

仔细看，由于 zebra 这个服务的任务主要是在修改 Linux 系统核心内的路由， 所以他仅监听本机接口而已，并不会监听外部的接口才对！另外，在 zebra.conf 这个档案当中，我们所设定的那个密码是有作用的喔！可以让我们登入 zebra 这套软件呢！ 好了，我们来查一查这个 2601 的 port 是否正确的启动的呢？

```
[root@www ~]# telnet localhost 2601
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.

Hello, this is Quagga (version 0.99.15).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

User Access Verification

Password: &lt;==在这里输入刚刚你设定的密码啊！
www.centos.vbird&gt; ? &lt;==在这边输入『 ? 』就能够知道有多少指令可使用
  echo      Echo a message back to the vty
  enable    Turn on privileged mode command
  exit      Exit current mode and down to previous mode
  help      Description of the interactive help system
  list      Print command list
  quit      Exit current mode and down to previous mode
  show      Show running system information
  terminal  Set terminal line parameters
  who       Display who is on vty
www.centos.vbird&gt; list &lt;==列出所有可用指令
  echo .MESSAGE
....(中间省略)....
  show debugging zebra
  show history
  show interface [IFNAME]
....(中间省略)....
  show ip protocol
  show ip route
....(其他省略)....
www.centos.vbird&gt; show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, &gt; - selected route, * - FIB route

K&gt;* 0.0.0.0/0 via 192.168.1.254, eth0            &lt;==核心直接设定的
C&gt;* 127.0.0.0/8 is directly connected, lo        &lt;==接口产生的路由
K&gt;* 169.254.0.0/16 is directly connected, eth1   &lt;==核心直接设定的
C&gt;* 192.168.1.0/24 is directly connected, eth0   &lt;==接口产生的路由
C&gt;* 192.168.100.0/24 is directly connected, eth1 &lt;==接口产生的路由
www.centos.vbird&gt; exit
Connection closed by foreign host.
```

仔细看到，我们登入这个 zebra 的服务之后，可以输入『help』或问号『?』， zebra 就会显示出你能够执行的指令有哪些，比较常用的当然是查询路由规则啰！以『 show ip route 』来查阅，结果可以发现目前的接口与默认路由都被显示出来了，显示的结果当中：

- K ：代表以类似 route 指令加入核心的路由规则，包括 route-ethN 所产生的规则；
- C ：代表由你的网络接口所设定的 IP 而产生的相关的路由规则
- S ：以 zebra 功能所设定的静态路由信息；
- R ：就是透过 RIP 协议所增加的路由规则啰！ 事实上，如果你还想要增加额外的静态路由的话，也可以透过 zebra 而不必使用 route 指令呢！ 例如想要增加 10.0.0.0/24 给 eth0 来处理的话，可以这样做：

```
[root@www ~]# vim /etc/quagga/zebra.conf
# 新增底下这一行喔！
ip route 10.0.0.0/24 eth0

[root@www ~]# /etc/init.d/zebra restart
[root@www ~]# telnet localhost 2601
Password: &lt;==这里输入密码
www.centos.vbird&gt; show ip route
K&gt;* 0.0.0.0/0 via 192.168.1.254, eth0
&lt;u&gt;S&gt;* 10.0.0.0/24 [1/0] is directly connected, eth0&lt;/u&gt;
C&gt;* 127.0.0.0/8 is directly connected, lo
K&gt;* 169.254.0.0/16 is directly connected, eth1
C&gt;* 192.168.1.0/24 is directly connected, eth0
C&gt;* 192.168.100.0/24 is directly connected, eth1
```

嘿嘿！立刻就会多出一笔路由的规则，而且最右边会显示 S，亦即是静态路由 (Static route) 的意思。 如此一来，我们系统管理员可就轻松多了！设定完右边 Router Z1 的 zebra 之后，不要忘记设定你的 Router Z2 呦！ 同样的设定再来一遍啦！只是主机名与密码应该给予不同才是呦！因为过程都一样，鸟哥就不再重复设定。 接下来我们可以开始看看 ripd 这个服务啰！

- 3. 在两部 Router 上面设定 ripd 服务：

ripd 这个服务可以在两部 Router 之间进行路由规则的交换与沟通，当然啦，如果你的环境里面有类似 Cisco 或者是其他有提供 RIP 协议的路由器的话，那么你当然也是可以透过这个 RIP 让你的 Linux Router 与其他硬件路由器互相沟通的呐！ 只不过 CentOS 6.x 的 quagga 所提供的 ripd 服务使用的是 RIPv2 版本，这个版本预设就要求得要进行身份验证的动作， 但是我们是个小型网络，并不想要加入这个身份验证的功能，因此就得要增加某些设定值才能够顺利的启动 ripd 呦！

先来设定 Router Z1 吧！在 Router Z1 当中，我们主要是透过 eth0 发送所有的网域路由信息，同时，我们管理的网域有 192.168.1.0/24, 192.168.100.0/24。再加上取消身份验证的设定值后，我们的 ripd 就会变成这样：

```
[root@www ~]# vim /etc/quagga/ripd.conf
hostname www.centos.vbird           &lt;==这里是设定 Router 的主机名而已
password linuxz1                    &lt;==设定好你自己的密码喔！
debug rip events                    &lt;==可以记录较多的错误讯息！
debug rip packet                    &lt;==鸟哥透过这个讯息解决很多问题
router rip                          &lt;==启动 Router 的 rip 功能
 version 2                          &lt;==启动的是 RIPv2 的服务 (默认值)
 network 192.168.1.0/24             &lt;==这两个就是我们管理的接口啰！
 network 192.168.100.0/24
&lt;u&gt;interface eth0&lt;/u&gt;                    &lt;==针对外部的那个接口，要略过身
 &lt;u&gt;no ip rip authentication mode&lt;/u&gt;      &lt;==就是这个项目！不要验证身份！
log file /var/log/quagga/zebra.log  &lt;==登录档设定与 zebra 相同即可

[root@www ~]# /etc/init.d/ripd start
[root@www ~]# chkconfig ripd on
[root@www ~]# netstat -tulnp &#124; grep ripd
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address   Foreign Address State   PID/Program name
tcp        0      0 127.0.0.1:2602 0.0.0.0:*       LISTEN  4456/ripd
udp        0      0 0.0.0.0:520    0.0.0.0:*               4456/ripd
# 新版的 quagga 启动的 2602 仅在 127.0.0.1，是透过 port 520 来传递信息！
```

基本上，这样就设定完成一部路由器的 RIP 动态路由协议了！在上头 ripd.conf 的设定当中， 他会主动以 eth0 及 192.168.1.0/24 这个网域的功能来进行搜索，如此一来，未来你进行任何路由规则的变动， 或者是整个网域的主机 IP 进行更动，你将不需要重新到每部 Router 上更动！ 因为这些路由器会自动的更新他们自己的规则喔！嘿嘿！接下来，同样的动作请你到 图 8.3-1 左边那部 Router Z2 上面设定一下！ 因为整个设定的流程都一样，所以这里鸟哥就省略啦！

- 4. 检查 RIP 协议的沟通结果：

在两部 Linux Router 都设定妥当之后，你可以登入 zebra 去看这两部主机的路由更新结果喔！ 举例来说，鸟哥登入图 8.3-1 右边那部 Router Z1 后，并且登入 zebra ， 观察路由会是这样的情况：

```
[root@www ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.100.0   0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.200.0   192.168.1.200   255.255.255.0   UG    2      0        0 eth0
0.0.0.0         192.168.1.254   0.0.0.0         UG    0      0        0 eth0
# 其实看路由就知道啦！那条有点线的就是新增的路由规则！很清楚！

[root@www ~]# telnet localhost 2601
Password: &lt;==不要忘记了密码啊！
www.centos.vbird&gt; show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, &gt; - selected route, * - FIB route

K&gt;* 0.0.0.0/0 via 192.168.1.254, eth0
S&gt;* 10.0.0.0/24 [1/0] is directly connected, eth0
C&gt;* 127.0.0.0/8 is directly connected, lo
K&gt;* 169.254.0.0/16 is directly connected, eth1
C&gt;* 192.168.1.0/24 is directly connected, eth0
C&gt;* 192.168.100.0/24 is directly connected, eth1
R&gt;* 192.168.200.0/24 [120/2] via 192.168.1.200, eth0, 00:02:43
```

如果你有看到上述的字体，嘿嘿！那就是成功啦！那个最左边的 R 代表的是透过 RIP 通讯协议所设定的路由规则啦！ 如此一来，咱们的路由器设定就搞定啰～如果一切都没有问题，你也想要开机就启动 zebra, ripd，那么还得要这样：

```
[root@www ~]# chkconfig zebra on
[root@www ~]# chkconfig ripd on
```

透过这个 quagga 以及 RIPv2 的路由协议的辅助，我们可以轻松的就将路由规则分享到附近区网的其他路由器上头， 比起单纯使用 route 去修改 Linux 的核心路由表，这个动作当然要快速很多！不过，如果是很小型的网络环境，那么不要使用这个 quagga 啊！因为有点多此一举的感觉。如果你的企业环境真的有够大，那么玩一玩这个 quagga 配合一些动态路由协议，嘿嘿！也是可行的啦！

Tips: 鸟哥差一点被这一版的 ripd.conf 设定内容搞死～因为 CentOS 5.x 以后的版本预设的 RIPv2 会去进行身份验证，所以原先在 CentOS 4.x 的设定是不能用的，偏偏登录档又看不出个所以然..后来查到可以透过 ripd.conf 内的 debug 参数去设定除错登录， 才发现 RIPv2 的认证问题！最终 google 一下才解决问题～好累啊！



8.4 特殊状况：路由器两边界面是同一个 IP 网段：ARP Proxy

如果你一开始设计的网络环境就是同一个 Class C 的网域，例如 192.168.1.0/24 ，后来因为某些因素必须要将某些主机搬到比较内部的环境中，例如图 8.2-1的 clientlinxu, winxp, win7。然后又因为某些因素，所以你不能变更这些计算机的 IP，此时你的同一网域就会横跨在一个路由器的左右两边了！ 举例来说，联机图示有点像底下这样：

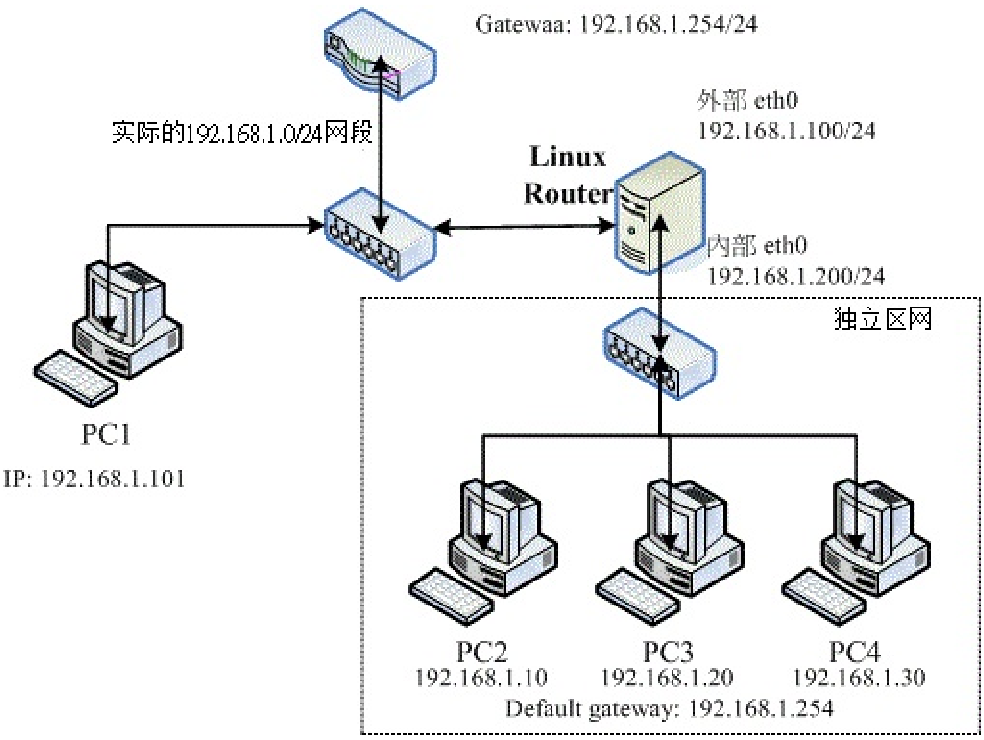


图 8.4-1、在路由器两个界面两边的 IP 是在同一个网域的设定情况

初次见面～看到眼睛快要掉下来哩！怎么路由器的两边的主机 IP 设定都在同一个网域内？而且还被规定不能够更改原先的 IP 设定， ...眞是一个头两个大啊～如此一来，在 Linux Router 两边要如何制作路由啊？好问题！眞是好问题～ 因为 OSI 第三层网络层的路由是一条一条去设定比对的，所以如果两块网卡上面都是同一个网域的 IP 时， 就会发生错误。那如何处理啊？

我们先从两方面来说，第一个，当从正确的网段 (PC1) 要联机到 PC2~PC4 时，他应该是要透过 Linux Router 那部主机的对外 IP (192.168.1.100) 才行！而且 Linux Router 还必须要让该封包透过内部 IP (192.168.1.200) 联机到 PC2~PC4 。 此时，封包传递的图示有点像这样：

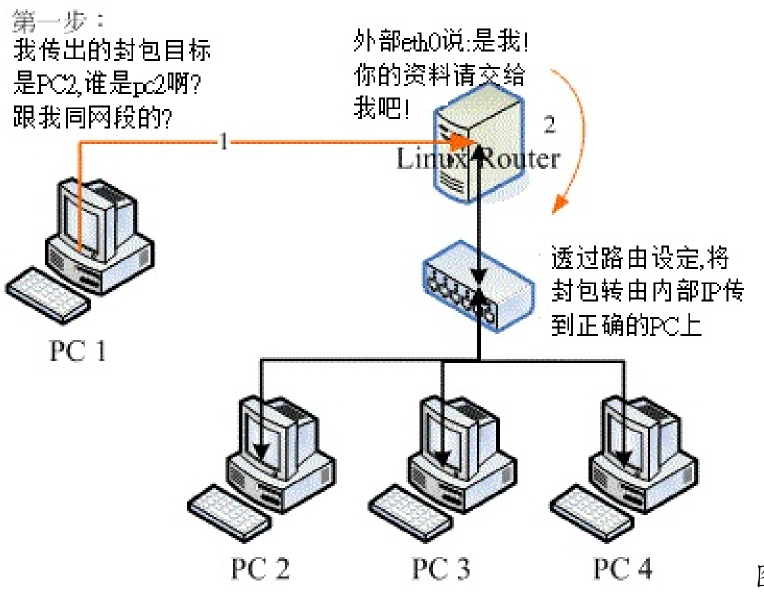


图 8.4-2、正常的网段想要传送到内部计算机去的封包流向

在这个阶段，我们可以设定PC2~PC4 的 IP 所对应的网卡卡号 (MAC) 都设定在 router 的对外网卡上， 因此， router 的对外接口可以将给 PC2~PC4 的封包给『骗』过来。接下来，就简单的透过路由设定，让封包转个接口发送出去即可。 这样 PC1 --> PC2 的问题解决了，但是 PC2 怎么传送到 PC1 呢？我们可以透过底下的图示来想象一下：

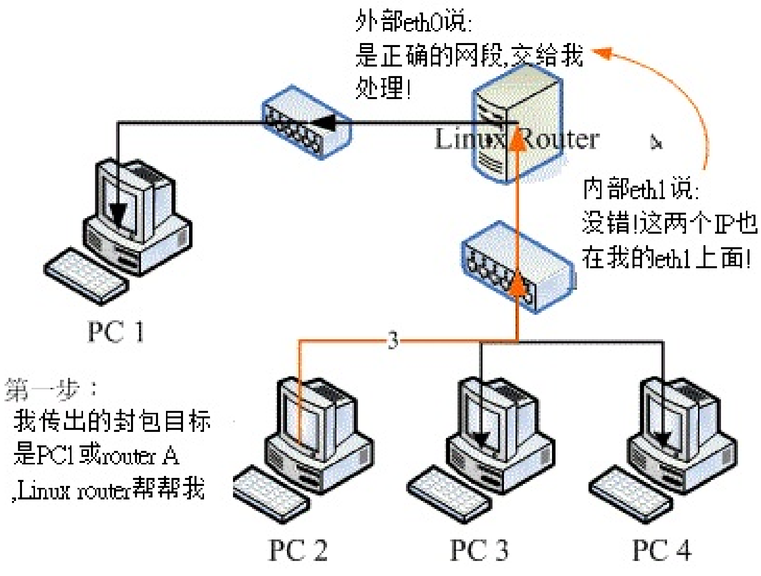


图 8.4-3、内部计算机想要传送到正常网域时的封包流向

当 PC2 要传送的封包是给 PC3, PC4 的，那么这个封包得要能够直接传递。但是如果需要传送到正常网域的封包，就得要透过 router 的对内网卡，再透过路由规则来将该封包导向外部接口来传递才行！这个时候就变成内部的接口欺骗 PC2 说， PC1 与 Router A 的 IP 是在内部这张接口上就是了，然后再透过路由判断将该封包透过外部接口来对外传递出去即可。 假设 Linux router 的对外界面为 eth0 而对内为 eth1 时，我们可以这样说：

1. 当 Linux Router 的 eth0 那个网域主机想要连接到 PC2~PC4 的主机时，由 Linux Router 负责接收；

2. 当 Linux Router 要传送数据到 PC2~PC4 时，务必要由 eth1 来传送；
3. 当内部计算机想要连接到 PC1 或 Router A 时，由 Linux router 的 eth1 负责接收；
4. 当 Linux Router 要传送的数据为 192.168.1.0/24 ，但并非 PC2~PC4 时，需由 eth0 传送。

上列的步骤与图示内的线条上的顺序相符合呦！得要对照着看看。其中的 (1) 与 (3) 就是透过 ARP Proxy (代理) 的功能啦！ 那啥是 ARP Proxy 呢？简单的说，就是让我的某张适配卡的 MAC 代理其他主机的 IP 对应，让想要连接到这个 IP 的 MAC 封包由我帮他接下来的意思。举我们图 8.4-1 的例子来说，就是在 Linux Router 的 eth0 界面上，规定 192.168.1.10, 192.168.1.20, 192.168.1.30 这三个 IP 都对应到 eth0 的 MAC 上，所以三个 IP 的封包就会由 eth0 代为收下，因此才叫做 ARP 代理人嘛！所以啦，每一部在 eth0 那端的主机都会『误判』那三个 IP 是 Linux Router 所拥有，这样就能够让封包传给 Linux Router 啦！

再接下来，咱们的 Linux Router 必须要额外指定路由，设定情况为：

- 若目标是 PC2 ~ PC4 时，该路由必须要由内部的 eth1 发送出去才行，
- 若目标不为 PC2 ~ PC4 ，且目标在 192.168.1.0/24 的网域时，需由 eth0 发送出去才行。

也就是说，你必须要指定路由规则当中，那个 PC2~PC4 具有优先选择权，然后其他的同网域封包才由 eth0 来传送。 这样就能够达成我们所想要的结局啦！^_^！看样子似乎很难，其实设定方面还挺简单的，你可以透过 arp 以及 route 这两个指令来达成喔！

- 外部接口 eth0：08:00:27:71:85:BD
- 内部接口 eth1：08:00:27:2A:30:14

```
# 1\. 先设定外部 eth0 的 ARP Proxy，让三个 IP 对应到自己的 MAC
[root@www ~]# arp -i eth0 -s 192.168.1.10 08:00:27:71:85:BD pub
[root@www ~]# arp -i eth0 -s 192.168.1.20 08:00:27:71:85:BD pub
[root@www ~]# arp -i eth0 -s 192.168.1.30 08:00:27:71:85:BD pub
[root@www ~]# arp -n
Address             HWtype  HWaddress      Flags Mask       Iface
192.168.1.30        *       *              MP               eth0
192.168.1.10        *       *              MP               eth0
192.168.1.20        *       *              MP               eth0
# 首先需要让外部接口拥有三个 IP 的操控权，透过这三个指令来建立 ARP 对应！

# 2\. 开始处理路由，增加 PC2~PC4 的单机路由经过内部的 eth1 来传递
[root@www ~]# route add -host 192.168.1.10 eth1
[root@www ~]# route add -host 192.168.1.20 eth1
[root@www ~]# route add -host 192.168.1.30 eth1
[root@www ~]# route -n
Kernel IP routing table
Destination    Gateway        Genmask         Flags Metric Ref    Use Iface
192.168.1.20   0.0.0.0        255.255.255.255 UH    0      0        0 eth1
192.168.1.10   0.0.0.0        255.255.255.255 UH    0      0        0 eth1
192.168.1.30   0.0.0.0        255.255.255.255 UH    0      0        0 eth1
192.168.1.0    0.0.0.0        255.255.255.0   U     0      0        0 eth0
192.168.1.0    0.0.0.0        255.255.255.0   U     0      0        0 eth1
0.0.0.0        192.168.1.254  0.0.0.0         UG    0      0        0 eth0
# 这样就处理好单向的单机路由啰！不过有个问题啊！那就是 192.168.1.0/24
# 的网域，两个接口都可以传送！因此，等一下第四个步骤得要将 eth1 删除才行！

3\. 设定一下内部的 ARP Proxy 工作 (绑在 eth1 上头啰)！
[root@www ~]# arp -i eth1 -s 192.168.1.101 08:00:27:2A:30:14 pub
[root@www ~]# arp -i eth1 -s 192.168.1.254 08:00:27:2A:30:14 pub
# 这样可以骗过 PC2 ~ PC4 ，让这三部主机传递的封包可以透过 router 来传递！

4\. 开始清除掉 eth1 的 192.168.1.0/24 路由
[root@www ~]# route del -net 192.168.1.0 netmask 255.255.255.0 eth1
```

所有的计算机都在同一个网域内，因此 default gatway 都是 192.168.1.254 ，而 netmask 都是 255.255.255.0， 只有 IP 不一样而已。最后，所有的计算机都可以直接跟对方联机，也能够顺利的连上 Internet ！ 这样的设定就能够满足上述的功能需求啰！如果一切都没有问题，那么将上述的指令写成一个脚本档， 例如 /root/bin/network.sh ，然后将该档案设定为可执行，并将它写入 /etc/rc.d/rc.local ， 同时每次重新启动网络后，就得要重新执行一次该脚本，即可达到你的需求啰！

透过这个案例你也可以清楚的知道，能不能联机其实与路由的关系才大哩！ 而路由是双向的，你必须要考虑到这个封包如何回来的问题喔！

8.5 重点回顾

- 网络卡的代号为 eth0, eth1, eth2...，而第一张网络卡的第一个虚拟接口为 eth0:0 ...
- 网络卡的参数可使用 ifconfig 直接设定，亦可使用配置文件如 /etc/sysconfig/network-scripts/ifcfg-ethn 来设定；
- 路由是双向的，所以由网络封包发送处发送到目标的路由规划，必须要考虑回程时是否具有相对的路由， 否则该封包可能会『遗失』；
- 每部主机都有自己的路由表，此路由表 (routing table) 是作为封包传送时的路径依据；
- 每部可对外 Internet 传送封包的主机，其路由信息中应有一个预设路由 (default gateway)；
- 要让 Linux 作为 Router 最重要的是启动核心的 IP Forward 功能；
- 重复路由可能会让你的网络封包传递到错误的方向；
- 动态路由通常是用在两个 Router 之间沟通彼此的路由规则用的，常见的 Linux 上的动态路由套件为 zebra ；
- arp proxy 可以透过 arp 与 route 的功能，让路由器两端都在同一个网段内；
- 一般来说，路由器上都会有两个以上的网络接口
- 事实上，Router 除了作为路由转换之外，在 Router 上面架设防火墙，亦可在企业内部再分隔出多个需要安全 (Security) 的单位数据的区隔！

8.6 本章习题

- 在练习完本章的相关信息后，请将网络环境还原成如图 3.2-1 的模样！ 以方便未来后续章节的练习喔！这个章节的路由器确实是有点麻烦的！^_^
- 请问你如何将你的 eth0 这个接口修改成为 192.168.100.2 在网域 192.168.100.0/25 之内的网络参数内容？因为 192.168.100.0/25 的 netmask 为 255.255.255.128 ，所以可以这样做： ifconfig eth0 192.168.100.2 netmask 255.255.255.128 up 这样即可！如果尚须其他的参数，则需要以档案形式来下达，如 vi /etc/sysconfig/network-scripts/ifcfg-eth0，并修改为： DEVICE=eth0 ONBOOT=yes BOOTPROTO=static IPADDR=192.168.100.2 NETMASK=255.255.255.128 NETWORK=192.168.100.0 BROADCAST=192.168.100.127
- 请手动设定 eth0:1 这个虚拟接口，使成为网络参数： 192.168.200.2, 网域在 192.168.200.0/24。ifconfig eth0:1 192.168.200.2 up
- 如何观察路由表？route -n 即可查阅！注意到 0.0.0.0 那个目标(default gateway)。
- 如何启动 Linux 的 IP Forward 功能？直接以『echo "1" > /proc/sys/net/ipv4/ip_forward 』即可！
- 假设你是一个学校单位的信息管理员，学校内有 200 部计算机，奉上面大头的旨意，必须要将 200 部计算机分为 4 个 Subnet ，请问你应该如何布线(请画出示意图)？而这 4 个 Subnet 的网络参数如何选择(请自行选择)？而是否需要 Router ？如果需要的话，假设每个 Router 仅能有两个网络实体接口，那么该如何布线？(注：不要使用虚拟接口)
- 假设你想要连接到 168.95.1.1 ，那么你该如何判断你经过『多少个』节点？可以使用 traceroute 168.95.1.1 来分析每个节点的传送信息，也可以透过 ping 168.95.1.1 所回传的那个 ttl 值判断节点数量。
- 万一你的网络有点停顿，发现可能是网络上某个节点出现问题，你应该如何确认是哪一部 Router 出问题？就利用 traceroute 吧！

8.7 参考数据与延伸阅读

- 注1：网中人写的『带宽负载平衡』：http://www.study-area.org/tips/multipath.htm
- 注2：一些生产网络硬设备的公司： 思科 (Cisco) ：http://www.cisco.com/ 友讯 (D-Link) ：http://www.dlinktw.com.tw/ 普联技术 (TP-Link) ：http://www.tplink.com.tw/
- 注3：动态路由架设软件： 动态路由软件 Quagga: http://www.quagga.net 动态路由软件 zebra: http://www.zebra.org Ben 哥写的『实作 Linux 动态路由』：http://linux.vbird.org/somepaper/20060714-linux_cisco_route.pdf quagga 官方操作文件：http://www.quagga.net/docs/quagga.pdf
- 酷学园的 ARP Proxy：http://phorum.study-area.org/viewtopic.php?t=5619
- 酷学园 ericshei 的 ARP Proxy 分享：http://phorum.study-area.org/viewtopic.php?t=22943

2002/08/09：第一次完成日期！ 2003/08/22：重新编辑文章，并增加重点回顾与课后练习 2006/08/21：将旧的文章移动到 此处。 2006/08/30：加入了 zebra 以及 ARP Proxy 等与 Router 比较相关的议题！ 2010/09/23：将旧的基于 CentOS 4.x 的版本移动到 此处 2010/10/26：修订了部分数据，尤其是 ARP 的说明部分！不过，重点于题目都还没有更新！ 2011/07/21：将基于 CentOS 5.x 的文章移动到此处 2011/07/22：将网域做成统一的格式，就是用第三章的区网架构来处理的喔！

第九章、防火墙与 NAT 服务器

最近更新日期：2011/07/22

从第七章的图 7.1-1 我们可以发现防火墙是整个封包要进入主机前的第一道关卡，但，什么是防火墙？Linux 的防火墙有哪些机制？ 防火墙可以达到与无法达到的功能有哪些？防火墙能不能作为区域防火墙而不是仅针对单一主机而已呢？其实，Linux 的防火墙主要是透过 Netfilter 与 TCP Wrappers 两个机制来管理的。其中，透过 Netfilter 防火墙机制，我们可以达到让私有 IP 的主机上网 (IP 分享器功能) ，并且也能够让 Internet 连到我内部的私有 IP 所架设的 Linux 服务器 (DNAT 功能)！真的很不赖喔！ 这一章对您来说，也真的有够重要的啦！

9.1 认识防火墙

网络安全除了随时注意相关软件的漏洞以及网络上的安全通报之外，你最好能够依据自己的环境来订定防火墙机制！ 这样对于你的网络环境，会比较有保障一点喔！那么什么是防火墙呢？其实防火墙就是透过订定一些有顺序的规则，并管制进入到我们网域内的主机 (或者可以说是网域) 数据封包的一种机制！更广义的来说，只要能够分析与过滤进出我们管理之网域的封包数据， 就可以称为防火墙。

防火墙又可以分为硬件防火墙与本机的软件防火墙。硬件防火墙是由厂商设计好的主机硬件， 这部硬件防火墙内的操作系统主要以提供封包数据的过滤机制为主，并将其他不必要的功能拿掉。因为单纯作为防火墙功能而已， 因此封包过滤的效率较佳。至于软件防火墙呢？那就是我们这个章节要来谈论的啊！ 软件防火墙本身就是在保护系统网络安全的一套软件(或称为机制)，例如 Netfilter 与 TCP Wrappers 都可以称为软件防火墙。

无论怎么分，反正防火墙就是用来保护我们网络安全的咚咚就对啦！我们这个章节主要在介绍 Linux 系统本身提供的软件防火墙的功能，那就是 Netfilter 。至于 TCP Wrappers 虽然在基础篇的第十八章认识系统服务里面谈过了，我们这里还会稍微简单的介绍啦！

9.1.1 开始之前来个提醒事项

由于本章主要的目的在介绍 Netfilter 这种封包过滤式的防火墙机制，因此网络基础里面的许多封包与讯框的概念要非常清楚， 包括网域的概念, IP 网域的撰写方式等，均需有一定的基础才行。请到第二章加强一下 MAC, IP, ICMP, TCP, UDP 等封包表头数据的认识，以及 Network/Netmask 的整体网域 (CIDR) 写法等。

另外，虽然 Netfilter 机制可以透过 iptables 指令的方式来进行规则的排序与修改，不过鸟哥建议你利用 shell script 来撰写属于你自己的防火墙机制比较好，因为对于规则的排序与汇整有比较好的观察性， 可以让你的防火墙规则比较清晰一点。所以在你开始了解底下的资料之前，希望你可以先阅读过相关的数据了：

- 已经认识 Shell 以及 Shell script；
- 已经阅读过第二章网络基础的内容；
- 已经阅读过第七章认识网络安全的内容；
- 已经阅读过第八章路由器的内容，了解重要的路由概念；
- 最好拥有两部主机以上的小型局域网络环境，以方便测试防火墙；
- 做为区域防火墙的 Linux 主机最好有两张实体网卡，可以进行多种测试，并架设 NAT 服务器；

9.1.2 为何需要防火墙

仔细分析第七章的图 7.1-1 可以发现， 封包进入本机时，会通过防火墙、服务器软件程序、SELinux与文件系统等。所以基本上，如果你的系统 (1)已经关闭不需要而且危险的服务； (2)已经将整个系统的所有软件都保持在最新的状态； (3)权限设定妥当且定时进行备份工作； (4)已经教育用户具有良好的网络、系统操作习惯。 那么你的系统实际上已经颇为安全了！要不要架设防火墙？那就见仁见智啰！

不过，毕竟网络世界是很复杂的，而 Linux 主机也不是一个简单的东西，说不定哪一天你在进行某个软件的测试时， 主机突然间就启动了一个网络服务，如果你没有管制该服务的使用范围，那么该服务就等于对所有 Internet 开放， 那就麻烦了！因为该服务可能可以允许任何人登入你的系统，那不是挺危险？

所以啰，防火墙能作什么呢？防火墙最大的功能就是帮助你『限制某些服务的存取来源』！ 举例来说： (1)你可以限制文件传输服务 (FTP) 只在子域内的主机才能够使用，而不对整个 Internet 开放； (2)你可以限制整部 Linux 主机仅可以接受客户端的 WWW 要求，其他的服务都关闭； (3)你还可以限制整部主机仅能主动对外联机。反过来说，若有客户端对我们主机发送主动联机的封包状态 (TCP 封包的 SYN flag) 就予以抵挡等等。这些就是最主要的防火墙功能了！

所以鸟哥认为，防火墙最重要的任务就是在规划出：

- 切割被信任(如子域)与不被信任(如 Internet)的网段；
- 划分出可提供 Internet 的服务与必须受保护的服务；
- 分析出可接受与不可接受的封包状态；

当然啦，咱们 Linux 的 iptables 防火墙软件还可以进行更细部深入的 NAT (Network Address Translation) 的设定，并进行更弹性的 IP 封包伪装功能，不过，对于单一主机的防火墙来说， 最简单的任务还是上面那三项就是了！所以，你需不需要防火墙呢？理论上，当然需要！ 而且你必须要知道『你的系统哪些数据与服务需要保护』，针对需要受保护的服务来设定防火墙的规则吧！ 底下我们先来谈一谈，那在 Linux 上头常见的防火墙类型有哪些？

9.1.3 Linux 系统上防火墙的主要类别

基本上，依据防火墙管理的范围，我们可以将防火墙区分为网域型与单一主机型的控管。在单一主机型的控管方面， 主要的防火墙有封包过滤型的 Netfilter 与依据服务软件程序作为分析的 TCP Wrappers 两种。若以区域型的防火墙而言， 由于此类防火墙都是当作路由器角色，因此防火墙类型主要则有封包过滤的 Netfilter 与利用代理服务器 (proxy server) 进行存取代理的方式了。

- Netfilter (封包过滤机制)

所谓的封包过滤，亦即是分析进入主机的网络封包，将封包的表头数据捉出来进行分析，以决定该联机为放行或抵挡的机制。 由于这种方式可以直接分析封包表头数据，所以包括硬件地址(MAC), 软件地址 (IP), TCP, UDP, ICMP 等封包的信息都可以进行过滤分析的功能，因此用途非常的广泛。(其实主要分析的是 OSI 七层协议的 2, 3, 4 层啦)

在 Linux 上面我们使用核心内建的 Netfilter 这个机制，而 Netfilter 提供了 iptables 这个软件来作为防火墙封包过滤的指令。由于 Netfilter 是核心内建的功能，因此他的效率非常的高！ 非常适合于一般小型环境的设定呢！Netfilter 利用一些封包过滤的规则设定，来定义出什么资料可以接收，什么数据需要剔除，以达到保护主机的目的喔！

- TCP Wrappers (程序控管)

另一种抵挡封包进入的方法，为透过服务器程序的外挂 (tcpd) 来处置的！与封包过滤不同的是， 这种机制主要是分析谁对某程序进行存取，然后透过规则去分析该服务器程序谁能够联机、谁不能联机。 由于主要是透过分析服务器程序来控管，因此与启动的埠口无关，只与程序的名称有关。 举例来说，我们知道 FTP 可以启动在非正规的 port 21 进行监听，当你透过 Linux 内建的 TCP wrappers 限制 FTP 时， 那么你只要知道 FTP 的软件名称 (vsftpd) ，然后对他作限制，则不管 FTP 启动在哪个埠口，都会被该规则管理的。

- Proxy (代理服务器)

其实代理服务器是一种网络服务，它可以『代理』用户的需求，而代为前往服务器取得相关的资料。就有点像底下这个图示吧：

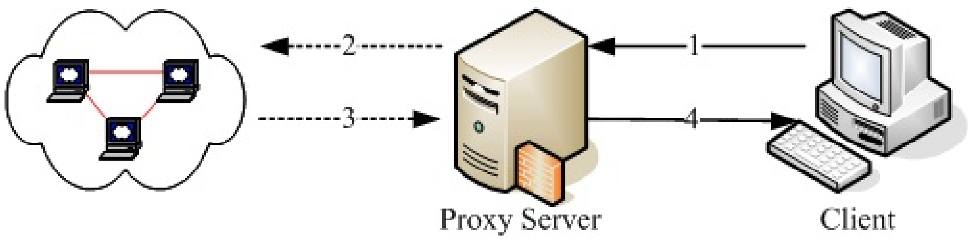


图 9.1-1、Proxy Server 的运作原理简介

```
以上图为例，当 Client 端想要前往 Internet 取得 Google 的数据时，他取得数据的流程是这样的：

1.  client 会向 proxy server 要求数据，请 proxy 帮忙处理；
2.  proxy 可以分析使用者的 IP 来源是否合法？使用者想要去的 Google 服务器是否合法？ 如果这个 client 的
3.  Google 所回传的数据是传给 proxy server 的喔，所以 Google 服务器上面看到的是 proxy server 的 I
4.  最后 proxy 将 Google 回传的数据送给 client。
这样了解了吗？没错， client 并没有直接连上 Internet ，所以在实线部分(步骤 1, 4)只要 Proxy 与 Client

另外，一般 proxy 主机通常仅开放 port 80, 21, 20 等 WWW 与 FTP 的埠口而已，而且通常 Proxy 就架设在路
```

9.1.4 防火墙的一般网络布线示意

由前面的说明当中，你应该可以了解到一件事，那就是防火墙除了可以『保护防火墙机制本身所在的那部主机』之外，还可以『保护防火墙后面的主机』。也就是说，防火墙除了可以防备本机被入侵之外，他还可以架设在路由器上面藉以控管进出本地端网域的网络封包。这种规划对于内部私有网域的安全也有一定程度的保护作用呢！底下我们稍微谈一谈目前常见的防火墙与网络布线的配置吧：

- 单一网域，仅有一个路由器：

  防火墙除了可以作为 Linux 本机的基本防护之外，他还可以架设在路由器上面以管控整个局域网络的封包进出。因此，在这类的防火墙上头通常至少需要有两个接口，将可信任的内部与不可信任的 Internet 分开，所以可以分别设定两块网络接口的防火墙规则啦！简单的环境如同下列图 9.1-2 所示。

  在图 9.1-2 中，由于防火墙是设定在所有网络封包都会经过的路由器上头，因此这个防火墙可以很轻易的就掌控到局域网络内的所有封包，而且你只要管理这部防火墙主机，就可以很轻易的将来自 Internet 的不良网络封包抵挡掉呐。只要管理一部主机就能够造福整的 LAN 里面的 PC，很划算的啦。

  如果你想要将局域网络控管的更严格的话，那你甚至可以在这部 Linux 防火墙上面架设更严格的代理服务器，让客户端仅能连上你所开放的 WWW 服务器而已，而且还可以透过代理服务器的登录文件分析功能，明确的查出来那个使用者在某个时间点曾经连上哪些 WWW 服务器，你瞧瞧！厉害吧！如果在这个防火墙上面再加装类似 MRTG 的流量监控软件，还能针对整个网域的流量进行监测。这样配置的优点是：

  - 因为内外网域已经分开，所以安全维护在内部可以开放的权限较大！
  - 安全机制的设定可以针对 Linux 防火墙主机来维护即可！
  - 对外只看的到 Linux 防火墙主机，所以对于内部可以达到有效的安全防护！

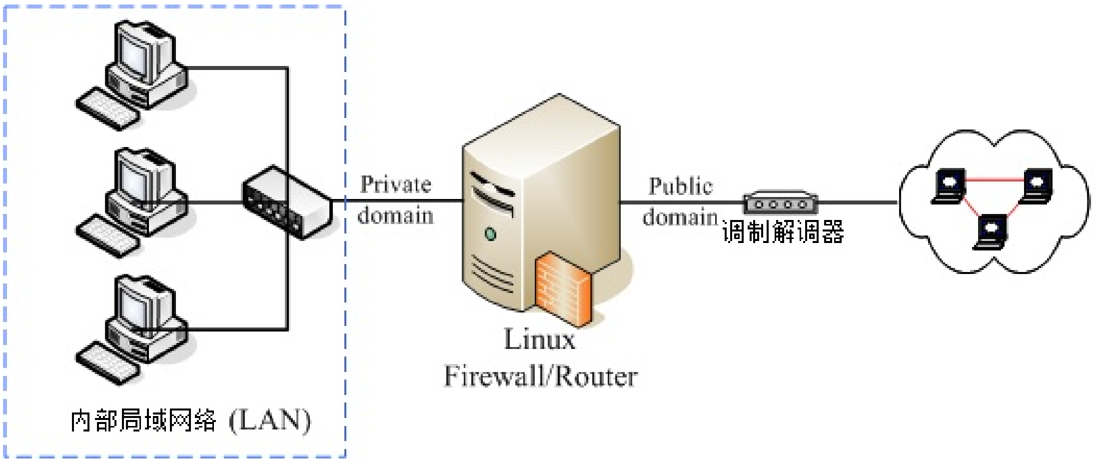


图 9.1-2、单一网域，仅有一个路由器的环境示意图

- 内部网络包含安全性更高的子网，需内部防火墙切开子网：

  一般来说，我们的防火墙对于 LAN 的防备都不会设定的很严格，因为是我们自己的 LAN 嘛！所以是信任网域之一啰！不过，最常听到的入侵方法也是使用这样的一个信任漏洞！因为你不能保证所有使用企业内部计算机的用户都是公司的员工，也无法保证你的

员工不会『搞破坏！』更多时候是由于某些外来访客利用移动式装置 (笔记本电脑) 连接到公司内部的无线网络来加以窃取企业内部的重要信息。

呵呵！所以，如果你有特别重要的部门需要更安全的保护网络环境，那么将 LAN 里面再加设一个防火墙，将安全等级分类，那么将会让你的重要数据获得更佳的保护喔！整个架构有点像下图所示。

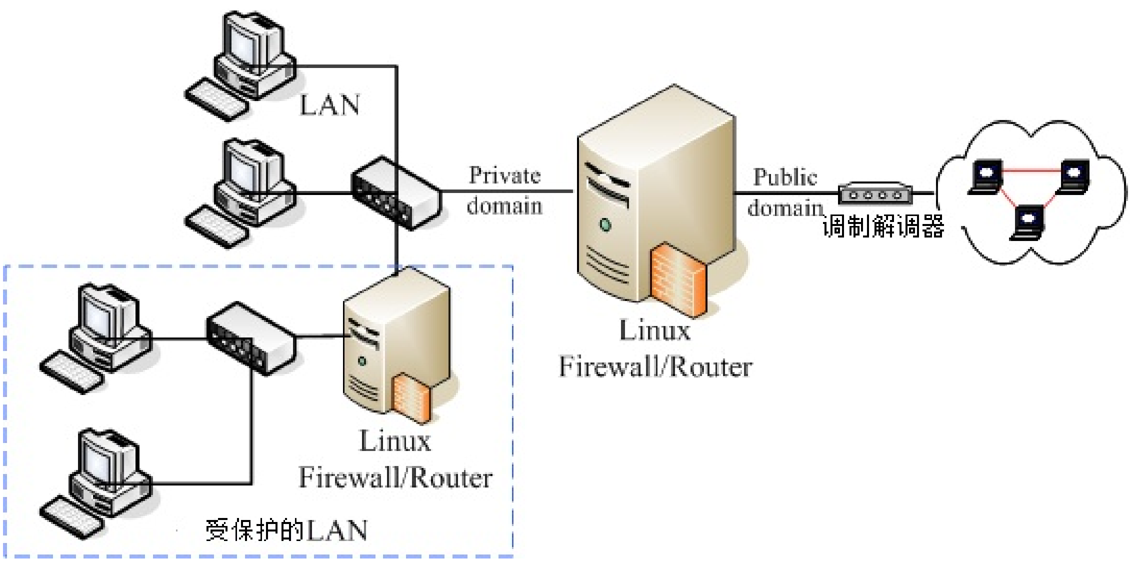


图 9.1-3、内部网络包含需要更安全的子网防火墙

- 在防火墙的后面架设网络服务器主机

还有一种更有趣的设定，那就是将提供网络服务的服务器放在防火墙后面，这有什么好处呢？ 如下图所示，Web, Mail 与 FTP 都是透过防火墙连到 Internet 上面去，所以， 底下这四部主机在 Internet 上面的 Public IP 都是一样的！(这个观念我们会在本章底下的 NAT 服务器的时候再次的强调)。 只是透过防火墙的封包分析后，将 WWW 的要求封包转送到 Web 主机，将 Mail 送给 Mail Server 去处理而已(透过 port 的不同来转递)。

好了，因为四部主机在 Internet 上面看到的 IP 都相同，但是事实上却是四部不同的主机， 而当有攻击者想要入侵你的 FTP 主机好了，他使用各种分析方法去进攻的主机，其实是『防火墙』那一部， 攻击者想要攻击你内部的主机，除非他能够成功的搞定你的防火墙，否则就很难入侵你的内部主机呢！

而且，由于主机放置在两部防火墙中间，内部网络如果发生状况时 (例如某些使用者不良操作导致中毒啊、 被社交工程攻陷导致内部主机被绑架啊等等的) ，是不会影响到网络服务器的正常运作的。 这种方式适用在比较大型的企业当中，因为对这些企业来说，网络主机能否提供正常稳定的服务是很重要的！

不过，这种架构下所进行的设定就得包含 port 的转递，而且要有很强的网络逻辑概念，可以厘清封包双向沟通时的流动方式。对于新手来说，设定上有一定的难度， 鸟哥个人不太建议新手这么做，还是等以后有经验之后再来玩这种架构吧！

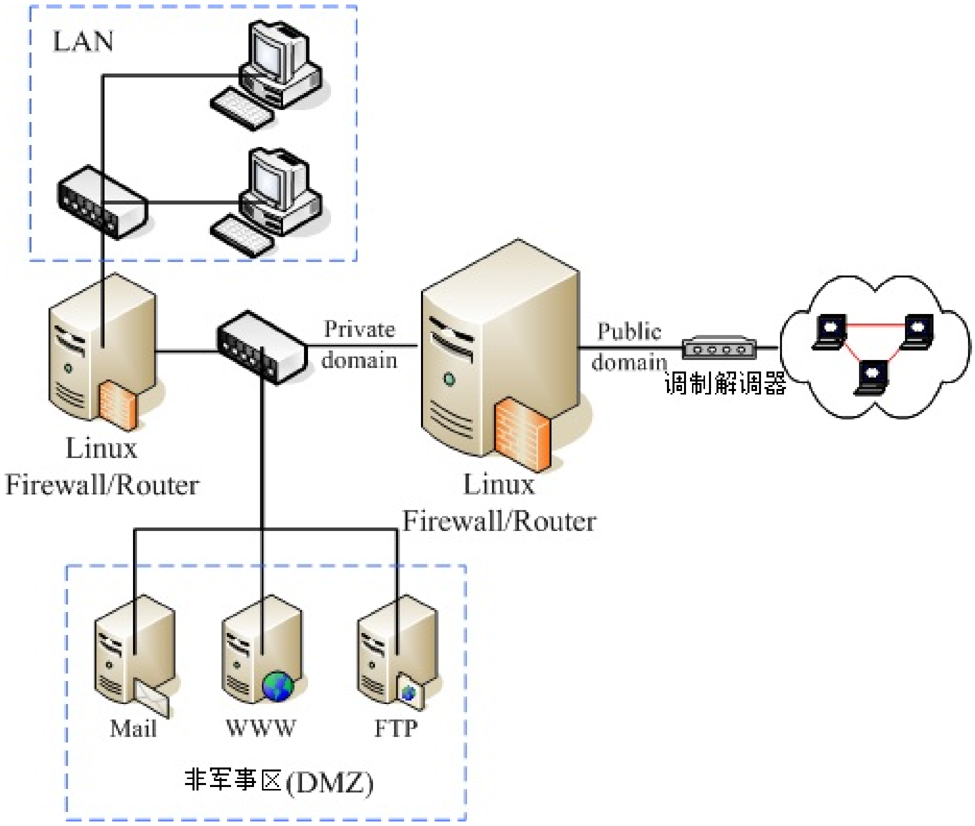


图 9.1-4、架设在防火墙后端的网络服务器环境示意图

通常像上图的环境中，将网络服务器独立放置在两个防火墙中间的网络，我们称之为非军事区域 (DMZ)。 DMZ 的目的就如同前面提到的，重点在保护服务器本身，所以将 Internet 与 LAN 都隔离开来，如此一来不论是服务器本身，或者是 LAN 被攻陷时，另一个区块还是完好无缺的！

9.1.5 防火墙的使用限制

从前面的分析中，我们已经知道过封包滤式防火墙主要在分析 OSI 七层协议当中的 2, 3, 4 层，既然如此的话， Linux 的 Netfilter 机制到底可以做些什么事情呢？其实可以进行的分析工作主要有：

- 拒绝让 Internet 的封包进入主机的某些端口 这个应该不难了解吧！例如你的 port 21 这个 FTP 相关的埠口，若只想要开放给内部网络的话，那么当 Internet 来的封包想要进入你的 port 21 时，就可以将该数据封包丢掉！因为我们可以分析的到该封包表头的端口号码呀！
- 拒绝让某些来源 IP 的封包进入 例如你已经发现某个 IP 主要都是来自攻击行为的主机，那么只要来自该 IP 的资料封包，就将他丢弃！这样也可以达到基础的安全哟！
- 拒绝让带有某些特殊旗标 (flag) 的封包进入 最常拒绝的就是带有 SYN 的主动联机的旗标了！只要一经发现，嘿嘿！你就可以将该封包丢弃呀！

- 分析硬件地址 (MAC) 来决定联机与否 如果你的局域网络里面有比较捣蛋的但是又具有比较高强的网络功力的高手时，如果你使用 IP 来抵挡他使用网络的权限，而他却懂得反正换一个 IP 就好了，都在同一个网域内嘛！ 同样还是在搞破坏～怎么办？没关系，我们可以死锁他的网络卡硬件地址啊！因为 MAC 是焊在网络卡上面的，所以你只要分析到该使用者所使用的 MAC 之后，可以利用防火墙将该 MAC 锁住，呵呵！除非他能够一换再换他的网络卡来取得新的 MAC，否则换 IP 是没有用的啦！

虽然 Netfilter 防火墙已经可以做到这么多的事情，不过，还是有很多事情没有办法透过 Netfilter 来完成喔！ 什么？设定防火墙之后还不安全啊！那当然啦！谁说设定了防火墙之后你的系统就一定安全？ 防火墙虽然可以防止不受欢迎的封包进入我们的网络当中，不过，某些情况下，他并不能保证我们的网络一定就很安全。 举几个例子来谈一谈：

- 防火墙并不能很有效的抵挡病毒或木马程序 假设你已经开放了 WWW 的服务，那么你的 WWW 主机上面，防火墙一定得要将 WWW 服务的 port 开放给 Client 端登入才行吧！否则你的 WWW 主机设定了等于没有用对吧！也就是说，只要进入你的主机的封包是要求 WWW 数据的，就可以通过你的防火墙。那好了，『万一你的 WWW 服务器软件有漏洞，或者本身向你要求 WWW 服务的该封包就是病毒在侦测你的系统』时，你的防火墙可是一点办法也没有啊！ 因为本来设定的规则就是会让他通过啊。
- 防火墙对于来自内部 LAN 的攻击较无承受力 一般来说，我们对于 LAN 里面的主机都没有什么防火墙的设定，因为是我们自己的 LAN 啊，所以当然就设定为信任网域了！不过， LAN 里面总是可能有些网络小白啊，虽然他们不是故意要搞破坏， 但是他们就是不懂嘛！所以就乱用网络了。这个时候就很糟糕，因为防火墙对于内部的规则设定通常比较少， 所以就容易造成内部员工对于网络误用或滥用的情况。

所以啦，还是回到第七章的图 7.1-1 的说明去看看，分析一下该图示，你就会知道，在你的 Linux 主机实地上网之前，还是得先：

- 关闭几个不安全的服务；
- 升级几个可能有问题的套件；
- 架设好最起码的安全防护--防火墙--

其他相关的讯息还是请到第七章认识网络安全里面去看一看怎么增加自身的安全吧！

9.2 TCP Wrappers

在进入主题之前，我们先来玩一个简单的防火墙机制，那就是 TCP Wrappers 这玩意儿。如同前面说的， TCP wrappers 是透过客户端想要链接的程序文件名，然后分析客户端的 IP ，看看是否需要放行。那么哪些程序支持 TCP wrappers 的功能？这个 TCP wrappers 又该如何设定？我们这里先简单的谈谈吧！(这个小节仅是简单的介绍过 TCP wrappers ，更多相关功能请参考基础学习篇的第十八章内容喔！)

9.2.1 哪些服务有支持

说穿了， TCP wrappers 就是透过 /etc/hosts.allow, /etc/hosts.deny 这两个宝贝蛋来管理的一个类似防火墙的机制， 但并非所有的软件都可以透过这两个档案来控管，只有底下的软件才能够透过这两个档案来管理防火墙规则，分别是：

- 由 super daemon (xinetd) 所管理的服务；
- 有支援 libwrap.so 模块的服务。

经由 xinetd 管理的服务还好理解，就是配置文件在 /etc/xinetd.d/ 里面的服务就是 xinetd 所管理的啊！ 那么什么是有支持 libwrap.so 模块呢？就让我们来进行底下的例题，你就比较容易明白啰：

例题：请查出你的系统有没有安装 xinetd ，若没有请安装。安装完毕后，请查询 xinetd 管理的服务有哪些？答：

```
[root@www ~]# yum install xinetd
Setting up Install Process
Package 2:xinetd-2.3.14-29.el6.x86_64 already installed and latest version
Nothing to do
# 画面中显示，已经是最新的 xinetd ！所以，已经有安装啰！
# 接下来找出 xinetd 所管理的服务群！

[root@www ~]# chkconfig xinetd on   &lt;==要先让 xinetd on 后才能看到底下的
[root@www ~]# chkconfig --list
....(前面省略)....
xinetd based services:
        chargen-dgram:  off
        chargen-stream: off
....(中间省略)....
        rsync:          off   &lt;==下一小节的范例就用这玩意儿来解释
        tcpmux-server:  off
        telnet:         on
```

上述结果最终输出的部分就是 xinetd 所管理的服务群啰！上述的服务之防火墙简易设定，都可以透过 TCP wrappers 来管理的噜！

例题：请问， rsyslogd, sshd, xinetd, httpd (若该服务不存在，请自行安装软件)，这四个程序有没有支持 tcp wrappers 的抵挡功能？答：由于支持 tcp wrappers 的服务必定包含 libwrap 这一个动态函式库，因此可以使用 ldd 来观察该服务即可。 简单的使用方式为：

```
[root@www ~]# ldd $(which rsyslogd sshd xinetd httpd)
# 这个方式可以将所有的动态函式库取出来查阅，不过需要眼睛搜寻。
# 可以透过底下的方式来处理更快！

[root@www ~]# for name in rsyslogd sshd xinetd httpd; do echo $name; \
&gt; ldd $(which $name) &#124; grep libwrap; done
rsyslogd
sshd
        libwrap.so.0 =&gt; /lib64/libwrap.so.0 (0x00007fb41d3c9000)
xinetd
        libwrap.so.0 =&gt; /lib64/libwrap.so.0 (0x00007f6314821000)
httpd
```

上述的结果中，在该档名档下有出现 libwrap 的，代表有找到该函式库，才有支持 tcp wrappers。 所以， sshd, xinetd 有支持，但是 rsyslogd, httpd 这两支程序则不支持。也就是说， httpd 与 rsyslogd 不能够使用 /etc/hosts.{allow|deny} 来进行防火墙机制的控管。

9.2.2 /etc/hosts.{allow|deny} 的设定方式

那如何透过这两个档案来抵挡有问题的 IP 来源呢？这两个档案的语法都一样，很简单的：

```
&lt;service(program_name)&gt; : &lt;IP, domain, hostname&gt;
&lt;服务   (亦即程序名称)&gt; : &lt;IP 或领域 或主机名&gt;
# 上头的 &gt; &lt; 是不存在于配置文件中的喔！
```

我们知道防火墙的规则都是有顺序的，那这两个档案与规则的顺序优先是怎样呢？基本上是这样的：

- 先以 /etc/hosts.allow 为优先比对，该规则符合就予以放行；
- 再以 /etc/hosts.deny 比对，规则符合就予以抵挡；
- 若不在这两个档案内，亦即规则都不符合，最终则予以放行。

我们拿 rsync 这个 xinetd 管理的服务来进行说明好了，请参考底下的例题吧：

例题：先开放本机的 127.0.0.1 可以进行任何本机的服务，然后，让区网 (192.168.1.0/24) 可以使用 rsync ， 同时 10.0.0.100 也能够使用 rsync ，但其他来源则不允许使用 rsync 喔。答：我们得要先知道 rsync 的服务启动的档名为何，因为 tcp wrappers 是透过启动服务的档名来管理的。 当我们观察 rsync 的配置文件时，可以发现：

```
[root@www ~]# cat /etc/xinetd.d/rsync
service rsync
{
        disable = yes
        flags           = IPv6
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/bin/rsync   &lt;==檔名叫做 rsync
        server_args     = --daemon
        log_on_failure  += USERID
}
```

因此程序字段的项目要写的是 rsync 喔！因此，我们应该要这样设定的：

```
[root@www ~]# vim /etc/hosts.allow
ALL: 127.0.0.1    &lt;==这就是本机全部的服务都接受！
rsync: 192.168.1.0/255.255.255.0 10.0.0.100

[root@www ~]# vim /etc/hosts.deny
rsync: ALL
```

上面的例题有几个重点，首先， tcp wrappers 理论上不支持 192.168.1.0/24 这种透过 bit 数值来定义的网域， 只支持 netmask 的地址显示方式。另外，如果有多个网域或者是单一来源，可以透过空格来累加。 如果想要写成多行呢？也可以啊！多写几行『 kshd: IP 』的方式也可以，不必要将所有数据集中在一行啦！因为 tcp wrappers 也是一条一条规则比对嘛！

基本上，你只要理解这些数据即可！因为绝大部分的时刻，我们都会建议使用底下介绍的 Netfilter 的机制来抵挡封包。 那让我们准备开始来玩玩 iptables 封包过滤防火墙吧！

9.3 Linux 的封包过滤软件：iptables

上面谈了这么多，主要还是希望你能了解到防火墙是什么这个议题！而且也希望你知道防火墙并非万能的。 好了，那么底下我们终于可以来瞧一瞧，那目前我们的 2.6 版这个 Linux 核心到底使用什么核心功能来进行防火墙设定？

9.3.1 不同 Linux 核心版本的防火墙软件

Linux 的防火墙为什么功能这么好？这是因为他本身就是由 Linux 核心所提供，由于直接经过核心来处理，因此效能非常好！ 不过，不同核心版本所使用的防火墙软件是不一样的！因为核心支持的防火墙是逐渐演进而来的嘛！

- Version 2.0：使用 ipfwadm 这个防火墙机制；
- Version 2.2：使用的是 ipchains 这个防火墙机制；
- Version 2.4 与 2.6 ：主要是使用 iptables 这个防火墙机制，不过在某些早期的 Version 2.4 版本的 distributions 当中，亦同时支持 ipchains (编译成为模块)，好让用户仍然可以使用来自 2.2 版的 ipchains 的防火墙规划。不过，不建议在 2.4 以上的核心版本使用 ipchains 喔！

因为不同的核心使用的防火墙机制不同，且支持的软件指令与语法也不相同，所以在 Linux 上头设定属于你自己的防火墙规则时，要注意啊，先用 uname -r 追踪一下你的核心版本再说！如果你是安装 2004 年以后推出的 distributions ，那就不需要担心了，因为这些 distributions 几乎都使用 kernel 2.6 版的核心啊！ ^_^

9.3.2 封包进入流程：规则顺序的重要性！

前面的几个小节里面我们一直谈到：『防火墙规则』，咦！啥是规则啊？因为 iptables 是利用封包过滤的机制， 所以他会分析封包的表头数据。根据表头数据与定义的『规则』来决定该封包是否可以进入主机或者是被丢弃。 意思就是说：『根据封包的分析资料 "比对" 你预先定义的规则内容， 若封包数据与规则内容相同则进行动作，否则就继续下一条规则的比对！』 重点在那个『比对与分析顺序』上。

举个简单的例子，假设我预先定义 10 条防火墙规则好了，那么当 Internet 来了一个封包想要进入我的主机， 那么防火墙是如何分析这个封包的呢？我们以底下的图示来说明好了：

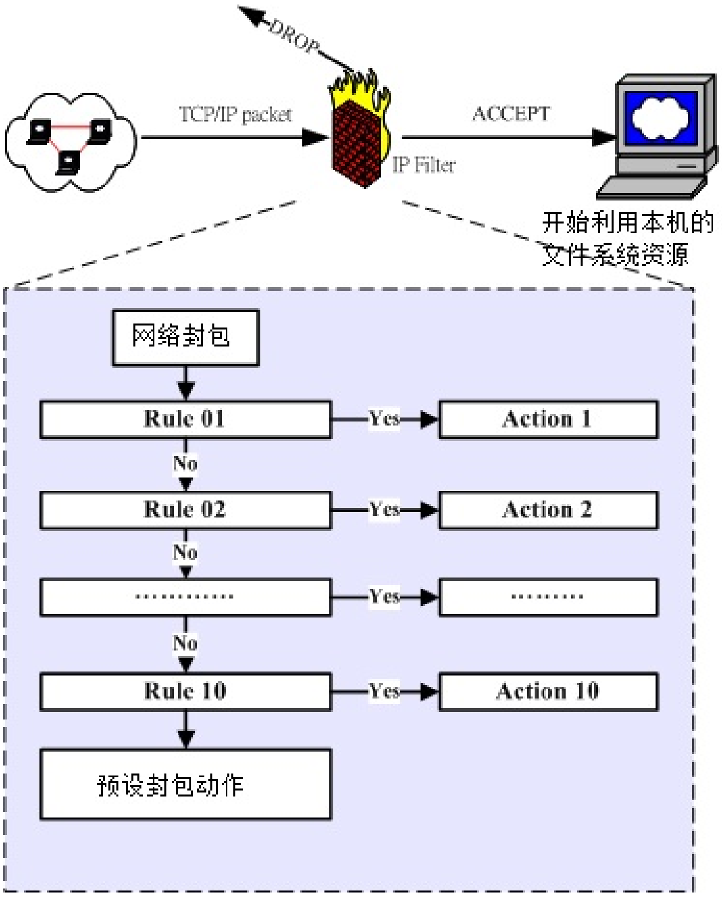


图 9.3-1、封包过滤的规则动作及分析流程

当一个网络封包要进入到主机之前，会先经由 NetFilter 进行检查，那就是 iptables 的规则了。 检查通过则接受 (ACCEPT) 进入本机取得资源，如果检查不通过，则可能予以丢弃 (DROP) ！ 上图中主要的目的在告知你：『规则是有顺序的』！例如当网络封包进入 Rule 1 的比对时， 如果比对结果符合 Rule 1 ，此时这个网络封包就会进行 Action 1 的动作，而不会理会后续的 Rule 2, Rule 3.... 等规则的分析了。

而如果这个封包并不符合 Rule 1 的比对，那就会进入 Rule 2 的比对了！如此一个一个规则去进行比对就是了。 那如果所有的规则都不符合怎办？此时就会透过预设动作 (封包政策, Policy) 来决定这个封包的去向。 所以啦，当你的规则顺序排列错误时，就会产生很严重的错误了。 怎么说呢？让我们看看底下这个例子：

假设你的 Linux 主机提供了 WWW 的服务，那么自然就要针对 port 80 来启用通过的封包规则，但是你发现 IP 来源为 192.168.100.100 老是恶意的尝试入侵你的系统，所以你想要将该 IP 拒绝往来，最后，所有的非 WWW 的封包都给他丢弃，就这三个规则来说，你要如何设定防火墙检验顺序呢？

1. Rule 1 先抵挡 192.168.100.100 ；
2. Rule 2 再让要求 WWW 服务的封包通过；
3. Rule 3 将所有的封包丢弃。

这样的排列顺序就能符合你的需求，不过，万一你的顺序排错了，变成：

1. Rule 1 先让要求 WWW 服务的封包通过；
2. Rule 2 再抵挡 192.168.100.100 ；
3. Rule 3 将所有的封包丢弃。

此时，那个 192.168.100.100 『可以使用你的 WWW 服务』喔！只要他对你的主机送出 WWW 要求封包，就可以使用你的 WWW 功能了，因为你的规则顺序定义第一条就会让他通过，而不去考虑第二条规则！这样可以理解规则顺序的意义了吗！ 现在再来想一想，如果 Rule 1 变成了『将所有的封包丢弃』，Rule 2 才设定『WWW 服务封包通过』，请问，我的 client 可以使用我的 WWW 服务吗？呵呵！答案是『否～』想通了吗？ ^_^

9.3.3 iptables 的表格 (table) 与链 (chain)

事实上，那个图 9.3-1 所列出的规则仅是 iptables 众多表格当中的一个链 (chain) 而已。 什么是链呢？这得由 iptables 的名称说起。为什么称为 ip"tables" 呢？ 因为这个防火墙软件里面有多个表格 (table) ，每个表格都定义出自己的默认政策与规则， 且每个表格的用途都不相同。我们可以使用底下这张图来稍微了解一下：

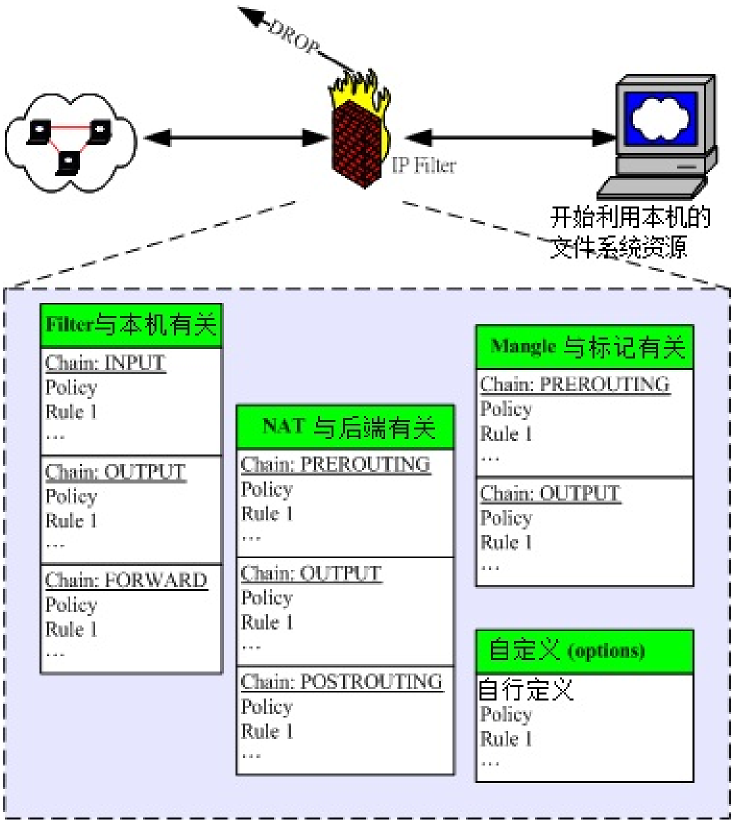


图 9.3-2、iptables 的表格与相关链示意图

刚刚图 9.3-1 的规则内容仅只是图 9.3-2 内的某个 chain 而已！ 而预设的情况下，咱们 Linux 的 iptables 至少就有三个表格，包括管理本机进出的 filter 、管理后端主机 (防火墙内部的其他计算机) 的 nat 、管理特殊旗标使用的 mangle (较少使用) 。更有甚者，我们还可以自定义额外的链呢！ 眞是很神奇吧！每个表格与其中链的用途分别是这样的：

- filter (过滤器)：主要跟进入 Linux 本机的封包有关，这个是预设的 table 喔！
    - INPUT：主要与想要进入我们 Linux 本机的封包有关；
    - OUTPUT：主要与我们 Linux 本机所要送出的封包有关；
    - FORWARD：这个咚咚与 Linux 本机比较没有关系， 他可以『转递封包』到后端的计算机中，与下列 nat table 相关性较高。
- nat (地址转换)：是 Network Address Translation 的缩写， 这个表格主要在进行来源与目的之 IP 或 port 的转换，与 Linux 本机较无关，主要与 Linux 主机后的局域网络内计算机较有相关。
    - PREROUTING：在进行路由判断之前所要进行的规则(DNAT/REDIRECT)
    - POSTROUTING：在进行路由判断之后所要进行的规则(SNAT/MASQUERADE)
    - OUTPUT：与发送出去的封包有关
- mangle (破坏者)：这个表格主要是与特殊的封包的路由旗标有关， 早期仅有 PREROUTING 及 OUTPUT 链，不过从 kernel 2.4.18 之后加入了 INPUT 及 FORWARD 链。 由于这个表格与特殊旗标相关性较高，所以像咱们这种单纯的环境当中，较少使用 mangle 这个表格。

所以说，如果你的 Linux 是作为 www 服务，那么要开放客户端对你的 www 要求有响应，就得要处理 filter 的 INPUT 链； 而如果你的 Linux 是作为局域网络的路由器，那么就得要分析 nat 的各个链以及 filter 的 FORWARD 链才行。也就是说， 其实各个表格的链结之间是有关系的！简单的关系可以由下图这么看：

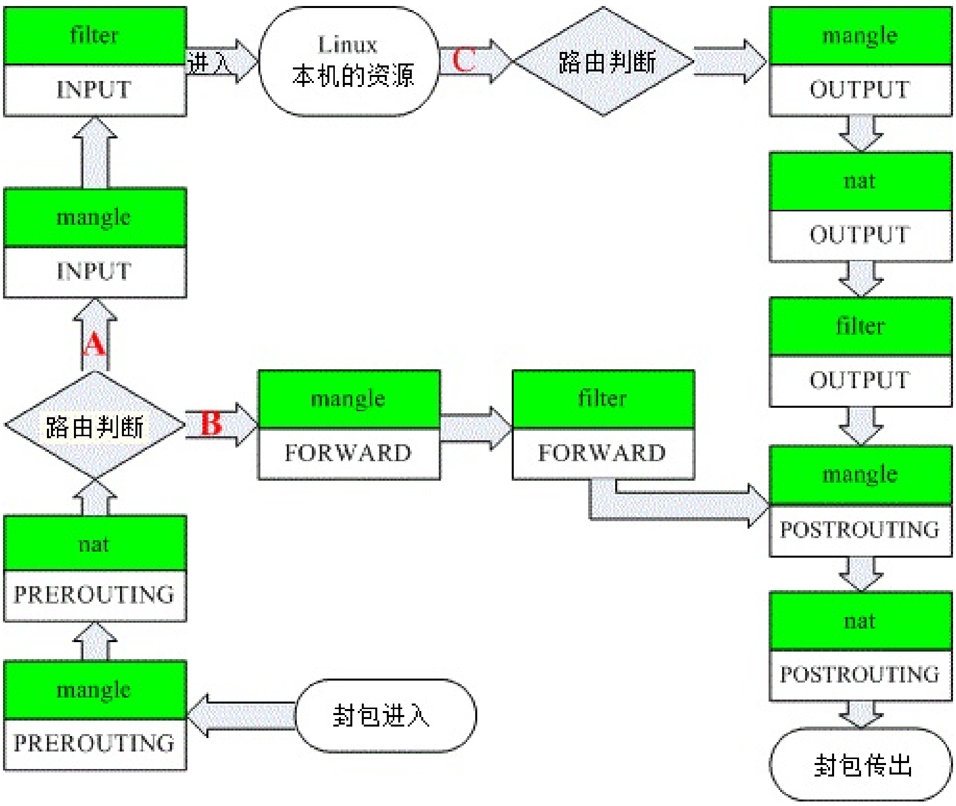


图 9.3-3、iptables 内建各表格与链的相关性

上面的图示很复杂喔！不过基本上你依旧可以看出来，我们的 iptables 可以控制三种封包的流向：

- 封包进入 Linux 主机使用资源 (路径 A)： 在路由判断后确定是向 Linux 主机要求数据的封包，主要就会透过 filter 的 INPUT 链来进行控管；
- 封包经由 Linux 主机的转递，没有使用主机资源，而是向后端主机流动 (路径 B)： 在路由判断之前进行封包表头的修订作业后，发现到封包主要是要透过防火墙而去后端，此时封包就会透过路径 B 来跑动。 也就是说，该封包的目标并非我们的 Linux 本机。主要经过的链是 filter 的 FORWARD 以及 nat 的 POSTROUTING, PREROUTING。 这路径 B 的封包流向使用情况，我们会在本章的 9.5 小节来跟大家作个简单的介绍。
- 封包由 Linux 本机发送出去 (路径 C)： 例如响应客户端的要求，或者是 Linux 本机主动送出的封包，都是透过路径 C 来跑的。先是透过路由判断， 决定了输出的路径后，再透过 filter 的 OUTPUT 链来传送的！当然，最终还是会经过 nat 的 POSTROUTING 链。

Tips: 有没有发现有两个『路由判断』呢？因为网络是双向的，所以进与出要分开来看！因此，进入的封包需要路由判断， 送出的封包当然也要进行路由判断才能够发送出去啊！了解乎？



由于 mangle 这个表格很少被使用，如果将图 9.3-3 的 mangle 拿掉的话，那就容易看的多了：

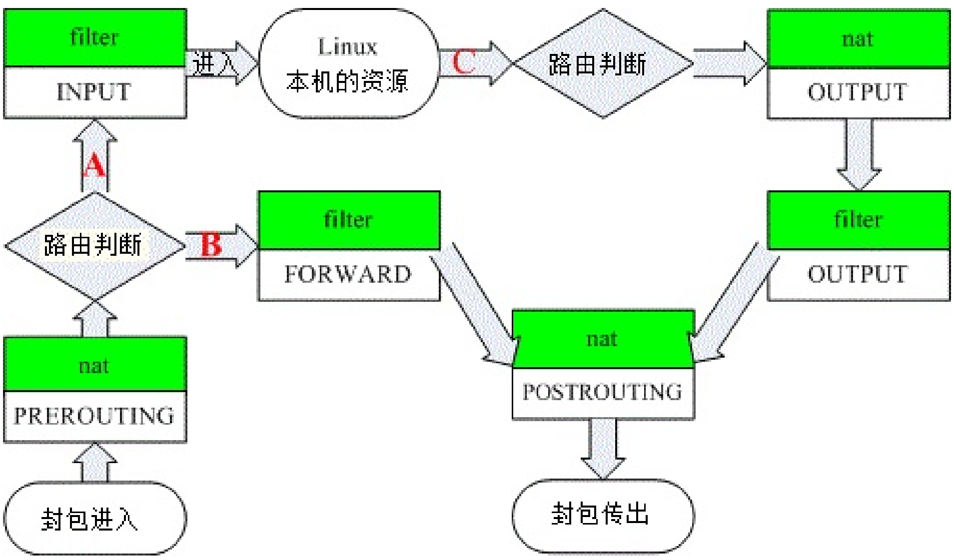


图 9.3-4、iptables 内建各表格与链的相关性(简图)

透过图 9.3-4 你就可以更轻松的了解到，事实上与本机最有关的其实是 filter 这个表格内的 INPUT 与 OUTPUT 这两条链，如果你的 iptables 只是用来保护 Linux 主机本身的话，那 nat 的规则根本就不需要理他，直接设定为开放即可。

不过，如果你的防火墙事实上是用来管制 LAN 内的其他主机的话，那么你就必须要再针对 filter 的 FORWARD 这条链，还有 nat 的 PREROUTING, POSTROUTING 以及 OUTPUT 进行额外的规则订定才行。 nat 表格的使用需要很清晰的路由概念才能够设定的好，建议新手先不要碰！最多就是先玩一玩最阳春的 nat 功能『IP 分享器的功能』就好了！ ^_^！这部份我们在本章的最后一小节会介绍的啦！

9.3.4 本机的 iptables 语法

理论上，当你安装好 Linux 之后，系统应该会主动的帮你启动一个阳春的防火墙规则才是，不过这个阳春防火墙可能不是我们想要的模式，因此我们需要额外进行一些修订的行为。不过，在开始进行底下的练习之前，鸟哥这里有个很重要的事情要告知一下。因为 iptables 的指令会将网络封包进行过滤及抵挡的动作，所以， 请不要在远程主机上进行防火墙的练习，因为你很有可能一不小心将自己关在家门外！ 尽量在本机前面登入 tty1-tty6 终端机进行练习，否则常常会发生悲剧啊！鸟哥以前刚刚在玩 iptables 时，就常常因为不小心规则设定错误，导致常常要请远程的朋友帮忙重新启动...

刚刚提到咱们的 iptables 至少有三个预设的 table (filter, nat, mangle)，较常用的是本机的 filter 表格， 这也是默认表格啦。另一个则是后端主机的 nat 表格，至于 mangle 较少使用，所以这个章节我们并不会讨论 mangle。 由于不同的 table 他们的链不一样，导致使用的指令语法或多或少都有点差异。 在这个小节当中，我们主要将针对 filter 这个默认表格的三条链来做介绍。底下就来玩一玩吧！

Tips: 防火墙的设定主要使用的就是 iptables 这个指令而已。而防火墙是系统管理员的主要任务之一， 且对于系统的影响相当的大，因此『只能让 root 使用 iptables 』，不论是设定还是观察防火墙规则喔！



9.3.4-1 规则的观察与清除

如果你在安装的时候选择没有防火墙的话，那么 iptables 在一开始的时候应该是没有规则的，不过， 可能因为你在安装的时候就有选择系统自动帮你建立防火墙机制，那系统就会有默认的防火墙规则了！ 无论如何，我们先来看看目前本机的防火墙规则是如何吧！

```
[root@www ~]# iptables [-t tables] [-L] [-nv]
选项与参数：
-t ：后面接 table ，例如 nat 或 filter ，若省略此项目，则使用默认的 filter
-L ：列出目前的 table 的规则
-n ：不进行 IP 与 HOSTNAME 的反查，显示讯息的速度会快很多！
-v ：列出更多的信息，包括通过该规则的封包总位数、相关的网络接口等

范例：列出 filter table 三条链的规则
[root@www ~]# iptables -L -n
Chain INPUT (policy ACCEPT)   &lt;==针对 INPUT 链，且预设政策为可接受
target  prot opt source     destination &lt;==说明栏
ACCEPT  all  --  0.0.0.0/0  0.0.0.0/0   state RELATED,ESTABLISHED &lt;==第 1 条规则
ACCEPT  icmp --  0.0.0.0/0  0.0.0.0/0                             &lt;==第 2 条规则
ACCEPT  all  --  0.0.0.0/0  0.0.0.0/0                             &lt;==第 3 条规则
ACCEPT  tcp  --  0.0.0.0/0  0.0.0.0/0   state NEW tcp dpt:22      &lt;==以下类推
REJECT  all  --  0.0.0.0/0  0.0.0.0/0   reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)  &lt;==针对 FORWARD 链，且预设政策为可接受
target  prot opt source     destination
REJECT  all  --  0.0.0.0/0  0.0.0.0/0   reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)  &lt;==针对 OUTPUT 链，且预设政策为可接受
target  prot opt source     destination

范例：列出 nat table 三条链的规则
[root@www ~]# iptables -t nat -L -n
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

在上表中，每一个 Chain 就是前面提到的每个链啰～ Chain 那一行里面括号的 policy 就是预设的政策， 那底下的 target, prot 代表什么呢？

- target：代表进行的动作， ACCEPT 是放行，而 REJECT 则是拒绝，此外，尚有 DROP (丢弃) 的项目！
- prot：代表使用的封包协议，主要有 tcp, udp 及 icmp 三种封包格式；
- opt：额外的选项说明
- source ：代表此规则是针对哪个『来源 IP』进行限制？
- destination ：代表此规则是针对哪个『目标 IP』进行限制？

在输出结果中，第一个范例因为没有加上 -t 的选项，所以默认就是 filter 这个表格内的 INPUT, OUTPUT, FORWARD 三条链的规则啰。若针对单机来说，INPUT 与 FORWARD 算是比较重要的管制防火墙链， 所以你可以发现最后一条规则的政策是 REJECT (拒绝) 喔！虽然 INPUT 与 FORWARD 的政策是放行 (ACCEPT)， 不过在最后一条规则就已经将全部的封包都拒绝了！

不过这个指令的观察只是作个格式化的查阅，要详细解释每个规则会比较不容易解析。举例来说， 我们将 INPUT 的 5 条规则依据输出结果来说明一下，结果会变成：

1. 只要是封包状态为 RELATED,ESTABLISHED 就予以接受
2. 只要封包协议是 icmp 类型的，就予以放行

3. 无论任何来源 (0.0.0.0/0) 且要去任何目标的封包，不论任何封包格式 (prot 为 all)，通通都接受
4. 只要是传给 port 22 的主动式联机 tcp 封包就接受
5. 全部的封包信息通通拒绝

最有趣的应该是第 3 条规则了，怎么会所有的封包信息都予以接受？如果都接受的话，那么后续的规则根本就不会有用嘛！ 其实那条规则是仅针对每部主机都有的内部循环测试网络 (lo) 接口啦！如果没有列出接口，那么我们就很容易搞错啰～ 所以，近来鸟哥都建议使用 iptables-save 这个指令来观察防火墙规则啦！因为 iptables-save 会列出完整的防火墙规则，只是并没有规格化输出而已。

```
[root@www ~]# iptables-save [-t table]
选项与参数：
-t ：可以仅针对某些表格来输出，例如仅针对 nat 或 filter 等等

[root@www ~]# iptables-save
# Generated by iptables-save v1.4.7 on Fri Jul 22 15:51:52 2011
*filter                       &lt;==星号开头的指的是表格，这里为 filter
:INPUT ACCEPT [0:0]           &lt;==冒号开头的指的是链，三条内建的链
:FORWARD ACCEPT [0:0]         &lt;==三条内建链的政策都是 ACCEPT 啰！
:OUTPUT ACCEPT [680:100461]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT &lt;==针对 INPUT 的规则
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT  &lt;==这条很重要！针对本机内部接口开放！
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited &lt;==针对 FORWARD 的规则
COMMIT
# Completed on Fri Jul 22 15:51:52 2011
```

由上面的输出来看，有底线且内容含有 lo 的那条规则当中，『 -i lo 』指的就是由 lo 适配卡进来的封包！ 这样看就清楚多了！因为有写到接口的关系啊！不像之前的 iptables -L -n 嘛！这样了解乎！ 不过，既然这个规则不是我们想要的，那该如何修改规则呢？鸟哥建议，先删除规则再慢慢建立各个需要的规则！ 那如何清除规则？这样做就对了：

```
[root@www ~]# iptables [-t tables] [-FXZ]
选项与参数：
-F ：清除所有的已订定的规则；
-X ：杀掉所有使用者 "自定义" 的 chain (应该说的是 tables ）啰；
-Z ：将所有的 chain 的计数与流量统计都归零

范例：清除本机防火墙 (filter) 的所有规则
[root@www ~]# iptables -F
[root@www ~]# iptables -X
[root@www ~]# iptables -Z
```

由于这三个指令会将本机防火墙的所有规则都清除，但却不会改变预设政策 (policy) ， 所以如果你不是在本机下达这三行指令时，很可能你会被自己挡在家门外 (若 INPUT 设定为 DROP 时)！要小心啊！

一般来说，我们在重新定义防火墙的时候，都会先将规则给他清除掉。还记得我们前面谈到的， 防火墙的『规则顺序』是有特殊意义的，所以啰， 当然先清除掉规则，然后一条一条来设定会比较容易一点啦。底下就来谈谈定义预设政策吧！

9.3.4-2 定义预设政策 (policy)

清除规则之后，再接下来就是要设定规则的政策啦！还记得政策指的是什么吗？『 当你的封包不在你设定的规则之内时，则该封包的通过与否，是以 Policy 的设定为准』，在本机方面的预设政策中，假设你对于内部的使用者有信心的话， 那么 filter 内的 INPUT 链方面可以定义的比较严格一点，而 FORWARD 与 OUTPUT 则可以订定的松一些！通常鸟哥都是将 INPUT 的 policy 定义为 DROP 啦，其他两个则定义为 ACCEPT。 至于 nat table 则暂时先不理会他。

```
[root@www ~]# iptables [-t nat] -P [INPUT,OUTPUT,FORWARD] [ACCEPT,DROP]
选项与参数：
-P ：定义政策( Policy )。注意，这个 P 为大写啊！
ACCEPT ：该封包可接受
DROP   ：该封包直接丢弃，不会让 client 端知道为何被丢弃。

范例：将本机的 INPUT 设定为 DROP ，其他设定为 ACCEPT
[root@www ~]# iptables -P INPUT   DROP
[root@www ~]# iptables -P OUTPUT  ACCEPT
[root@www ~]# iptables -P FORWARD ACCEPT
[root@www ~]# iptables-save
# Generated by iptables-save v1.4.7 on Fri Jul 22 15:56:34 2011
*filter
:INPUT DROP [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
COMMIT
# Completed on Fri Jul 22 15:56:34 2011
# 由于 INPUT 设定为 DROP 而又尚未有任何规则，所以上面的输出结果显示：
# 所有的封包都无法进入你的主机！是不通的防火墙设定！(网络联机是双向的)
```

看到输出的结果了吧？INPUT 被修改了设定喔！其他的 nat table 三条链的预设政策设定也是一样的方式，例如：『 iptables -t nat -P PREROUTING ACCEPT 』就设定了 nat table 的 PREROUTING 链为可接受的意思！预设政策设定完毕后，来谈一谈关于各规则的封包基础比对设定吧。

9.3.4-3 封包的基础比对：IP, 网域及接口装置

开始来进行防火墙规则的封包比对设定吧！既然是因特网，那么我们就由最基础的 IP, 网域及埠口，亦即是 OSI 的第三层谈起，再来谈谈装置 (网络卡) 的限制等等。这一小节与下一小节的语法你一定要记住，因为这是最基础的比对语法喔！

```
[root@www ~]# iptables [-AI 链名] [-io 网络接口] [-p 协议] \
&gt; [-s 来源IP/网域] [-d 目标IP/网域] -j [ACCEPT&#124;DROP&#124;REJECT&#124;LOG]
选项与参数：
-AI 链名：针对某的链进行规则的 "插入" 或 "累加"
    -A ：新增加一条规则，该规则增加在原本规则的最后面。例如原本已经有四条规则，
         使用 -A 就可以加上第五条规则！
    -I ：插入一条规则。如果没有指定此规则的顺序，默认是插入变成第一条规则。
         例如原本有四条规则，使用 -I 则该规则变成第一条，而原本四条变成 2~5 号
    链 ：有 INPUT, OUTPUT, FORWARD 等，此链名称又与 -io 有关，请看底下。

-io 网络接口：设定封包进出的接口规范
    -i ：封包所进入的那个网络接口，例如 eth0, lo 等接口。需与 INPUT 链配合；
    -o ：封包所传出的那个网络接口，需与 OUTPUT 链配合；

-p 协定：设定此规则适用于哪种封包格式
   主要的封包格式有： tcp, udp, icmp 及 all 。

-s 来源 IP/网域：设定此规则之封包的来源项目，可指定单纯的 IP 或包括网域，例如：
   IP  ：192.168.0.100
   网域：192.168.0.0/24, 192.168.0.0/255.255.255.0 均可。
   若规范为『不许』时，则加上 ! 即可，例如：
   -s ! 192.168.100.0/24 表示不许 192.168.100.0/24 之封包来源；

-d 目标 IP/网域：同 -s ，只不过这里指的是目标的 IP 或网域。

-j ：后面接动作，主要的动作有接受(ACCEPT)、丢弃(DROP)、拒绝(REJECT)及记录(LOG)
```

iptables 的基本参数就如同上面所示的，仅只谈到 IP 、网域与装置等等的信息， 至于 TCP, UDP 封包特有的埠口 (port number) 与状态 (如 SYN 旗标) 则在下小节才会谈到。 好，先让我们来看看最基础的几个规则，例如开放 lo 这个本机的接口以及某个 IP 来源吧！

```
范例：设定 lo 成为受信任的装置，亦即进出 lo 的封包都予以接受
[root@www ~]# iptables -A INPUT -i lo -j ACCEPT
```

仔细看上面并没有列出 -s, -d 等等的规则，这表示：不论封包来自何处或去到哪里，只要是来自 lo 这个界面，就予以接受！这个观念挺重要的，就是『没有指定的项目，则表示该项目完全接受』的意思！ 例如这个案例当中，关于 -s, -d...等等的参数没有规定时，就代表不论什么值都会被接受啰。

这就是所谓的信任装置啦！假如你的主机有两张以太网络卡，其中一张是对内部的网域，假设该网卡的代号为 eth1 好了， 如果内部网域是可信任的，那么该网卡的进出封包就通通会被接受，那你就能够用：『iptables -A INPUT -i eth1 -j ACCEPT』 来将该装置设定为信任装置。不过，下达这个指令前要特别注意，因为这样等于该网卡没有任何防备了喔！

```
范例：只要是来自内网的 (192.168.100.0/24) 的封包通通接受
[root@www ~]# iptables -A INPUT -i eth1 -s 192.168.100.0/24 -j ACCEPT
# 由于是内网就接受，因此也可以称之为『信任网域』啰。

范例：只要是来自 192.168.100.10 就接受，但 192.168.100.230 这个恶意来源就丢弃
[root@www ~]# iptables -A INPUT -i eth1 -s 192.168.100.10 -j ACCEPT
[root@www ~]# iptables -A INPUT -i eth1 -s 192.168.100.230 -j DROP
# 针对单一 IP 来源，可视为信任主机或者是不信任的恶意来源喔！

[root@www ~]# iptables-save
# Generated by iptables-save v1.4.7 on Fri Jul 22 16:00:43 2011
*filter
:INPUT DROP [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [17:1724]
-A INPUT -i lo -j ACCEPT
-A INPUT -s 192.168.100.0/24 -i eth1 -j ACCEPT
-A INPUT -s 192.168.100.10/32 -i eth1 -j ACCEPT
-A INPUT -s 192.168.100.230/32 -i eth1 -j DROP
COMMIT
# Completed on Fri Jul 22 16:00:43 2011
```

这就是最单纯简单的防火墙规则的设定与观察方式。不过，在上面的案例中，其实你也发现到有两条规则可能有问题～那就是上面的特殊字体圈起来的规则顺序。明明已经放行了 192.168.100.0/24 了，所以那个 192.168.100.230 的规则就不可能会被用到！这就是有问题的防火墙设定啊！了解乎？那该怎办？就重打啊！@_@！ 那如果你想要记录某个规则的纪录怎么办？可以这样做：

```
[root@www ~]# iptables -A INPUT -s 192.168.2.200 -j LOG
[root@www ~]# iptables -L -n
target prot opt source         destination
LOG    all  --  192.168.2.200  0.0.0.0/0    LOG flags 0 level 4
```

看到输出结果的最左边，会出现的是 LOG 喔！只要有封包来自 192.168.2.200 这个 IP 时，那么该封包的相关信息就会被写入到核心讯息，亦即是 /var/log/messages 这个档案当中。然后该封包会继续进行后续的规则比对。所以说， LOG 这个动作仅在进行记录而已，并不会影响到这个封包的其他规则比对的。 好了，接下来我们分别来看看 TCP,UDP 以及 ICMP 封包的其他规则比对吧！

9.3.4-4 TCP, UDP 的规则比对：针对埠口设定

我们在第二章网络基础谈过各种不同的封包格式， 在谈到 TCP 与 UDP 时，比较特殊的就是那个埠口 (port)，在 TCP 方面则另外有所谓的联机封包状态， 包括最常见的 SYN 主动联机的封包格式。那么如何针对这两种封包格式进行防火墙规则的设定呢？你可以这样看：

```
[root@www ~]# iptables [-AI 链] [-io 网络接口] [-p tcp,udp] \
&gt; [-s 来源IP/网域] [--sport 埠口范围] \
&gt; [-d 目标IP/网域] [--dport 埠口范围] -j [ACCEPT&#124;DROP&#124;REJECT]
选项与参数：
--sport 埠口范围：限制来源的端口号码，端口号码可以是连续的，例如 1024:65535
--dport 埠口范围：限制目标的端口号码。
```

事实上就是多了那个 --sport 及 --dport 这两个玩意儿，重点在那个 port 上面啦！ 不过你得要特别注意，因为仅有 tcp 与 udp 封包具有埠口，因此你想要使用 --dport, --sport 时，得要加上 -p tcp 或 -p udp 的参数才会成功喔！底下让我们来进行几个小测试：

```
范例：想要联机进入本机 port 21 的封包都抵挡掉：
[root@www ~]# iptables -A INPUT -i eth0 -p tcp --dport 21 -j DROP

范例：想连到我这部主机的网芳 (upd port 137,138 tcp port 139,445) 就放行
[root@www ~]# iptables -A INPUT -i eth0 -p udp --dport 137:138 -j ACCEPT
[root@www ~]# iptables -A INPUT -i eth0 -p tcp --dport 139 -j ACCEPT
[root@www ~]# iptables -A INPUT -i eth0 -p tcp --dport 445 -j ACCEPT
```

瞧！你可以利用 UDP 与 TCP 协议所拥有的端口号码来进行某些服务的开放或关闭喔！你还可以综合处理呢！例如：只要来自 192.168.1.0/24 的 1024:65535 埠口的封包，且想要联机到本机的 ssh port 就予以抵挡，可以这样做：

```
[root@www ~]# iptables -A INPUT -i eth0 -p tcp -s 192.168.1.0/24 \
&gt; --sport 1024:65534 --dport ssh -j DROP
```

如果忘记加上 -p tcp 就使用了 --dport 时，会发生啥问题呢？

```
[root@www ~]# iptables -A INPUT -i eth0 --dport 21 -j DROP
iptables v1.4.7: unknown option `--dport'
Try `iptables -h' or 'iptables --help' for more information.
```

你应该会觉得很奇怪，怎么『 --dport 』会是未知的参数 (arg) 呢？这是因为你没有加上 -p tcp 或 -p udp 的缘故啊！很重要喔！

除了埠口之外，在 TCP 还有特殊的旗标啊！最常见的就是那个主动联机的 SYN 旗标了。 我们在 iptables 里面还支持『 --syn 』的处理方式，我们以底下的例子来说明好了：

```
范例：将来自任何地方来源 port 1:1023 的主动联机到本机端的 1:1023 联机丢弃
[root@www ~]# iptables -A INPUT -i eth0 -p tcp --sport 1:1023 \
&gt; --dport 1:1023 --syn -j DROP
```

一般来说，client 端启用的 port 都是大于 1024 以上的埠口，而 server 端则是启用小于 1023 以下的埠口在监听的。所以我们可以让来自远程的小于 1023 以下的端口数据的主动联机都给他丢弃！ 但不适用在 FTP 的主动联机中！这部份我们未来在二十一章的 FTP 服务器再来谈吧！

9.3.4-5 iptables 外挂模块：mac 与 state

在 kernel 2.2 以前使用 ipchains 管理防火墙时，通常会让系统管理员相当头痛！因为 ipchains 没有所谓的封包状态模块，因此我们必须要针对封包的进、出方向进行管控。举例来说，如果你想要联机到远程主机的 port 22 时，你必须要针对两条规则来设定：

- 本机端的 1024:65535 到远程的 port 22 必须要放行 (OUTPUT 链)；
- 远程主机 port 22 到本机的 1024:65535 必须放行 (INPUT 链)；

这会很麻烦！因为如果你要联机到 10 部主机的 port 22 时，假设 OUTPUT 为预设开启 (ACCEPT)， 你依旧需要填写十行规则，让那十部远程主机的 port 22 可以联机到你的本地端主机上。 那如果开启全部的 port 22 呢？又担心某些恶意主机会主动以 port 22 联机到你的机器上！ 同样的道理，如果你要让本地端主机可以连到外部的 port 80 (WWW 服务)，那就更不得了～ 这就是网络联机是双向的一个很重要的概念！

好在我们的 iptables 免除了这个困扰！他可以透过一个状态模块来分析 『这个想要进入的封包是否为刚刚我发出去的响应？』 如果是刚刚我发出去的响应，那么就可以予以接受放行！ 哇！真棒！这样就不用管远程主机是否联机进来的问题了！ 那如何达到呢？看看底下的语法：

```
[root@www ~]# iptables -A INPUT [-m state] [--state 状态]
选项与参数：
-m ：一些 iptables 的外挂模块，主要常见的有：
     state ：状态模块
     mac   ：网络卡硬件地址 (hardware address)
--state ：一些封包的状态，主要有：
     INVALID    ：无效的封包，例如数据破损的封包状态
     ESTABLISHED：已经联机成功的联机状态；
     NEW        ：想要新建立联机的封包状态；
     RELATED    ：这个最常用！表示这个封包是与我们主机发送出去的封包有关

范例：只要已建立或相关封包就予以通过，只要是不合法封包就丢弃
[root@www ~]# iptables -A INPUT -m state \
&gt; --state RELATED,ESTABLISHED -j ACCEPT
[root@www ~]# iptables -A INPUT -m state --state INVALID -j DROP
```

如此一来，我们的 iptables 就会主动分析出该封包是否为响应状态，若是的话，就直接予以接受。呵呵！ 这样一来你就不需要针对响应的封包来撰写个别的防火墙规则了！这真是太棒了！底下我们继续谈一下 iptables 的另一个外挂， 那就是针对网卡来进行放行与防御：

```
范例：针对局域网络内的 aa:bb:cc:dd:ee:ff 主机开放其联机
[root@www ~]# iptables -A INPUT -m mac --mac-source aa:bb:cc:dd:ee:ff \
&gt; -j ACCEPT
选项与参数：
--mac-source ：就是来源主机的 MAC 啦！
```

如果你的区网当中有某些网络高手，老是可以透过修改 IP 去尝试透过路由器往外跑，那你该怎么办？ 难道将整个区网拒绝？并不需要的，你可以透过之前谈到的 ARP 相关概念，去捉到那部主机的 MAC ，然后透过上头的这个机制， 将该主机整个 DROP 掉即可。不管他改了什么 IP ，除非他知道你是用网卡的 MAC 来管理，否则他就是出不去啦！了解乎？

Tips: 其实 MAC 也是可以伪装的，可以透过某些软件来修改网卡的 MAC。不过，这里我们是假设 MAC 是无法修改的情况来说明的。 此外，MAC 是不能跨路由的，因此上述的案例中才特别说明是在区网内，而不是指 Internet 外部的来源喔！



9.3.4-6 ICMP 封包规则的比对：针对是否响应 ping 来设计

在第二章 ICMP 协议当中我们知道 ICMP 的类型相当的多，而且很多 ICMP 封包的类型都是为了要用来进行网络检测用的！所以最好不要将所有的 ICMP 封包都丢弃！如果不是做为路由器的主机时，通常我们会把 ICMP type 8 (echo request) 拿掉而已，让远程主机不知道我们是否存在，也不会接受 ping 的响应就是了。ICMP 封包格式的处理是这样的：

```
[root@www ~]# iptables -A INPUT [-p icmp] [--icmp-type 类型] -j ACCEPT
选项与参数：
--icmp-type ：后面必须要接 ICMP 的封包类型，也可以使用代号，
              例如 8  代表 echo request 的意思。

范例：让 0,3,4,11,12,14,16,18 的 ICMP type 可以进入本机：
[root@www ~]# vi somefile
#!/bin/bash
icmp_type="0 3 4 11 12 14 16 18"
for typeicmp in $icmp_type
do
   iptables -A INPUT -i eth0 -p icmp --icmp-type $typeicmp -j ACCEPT
done

[root@www ~]# sh  somefile
```

这样就能够开放部分的 ICMP 封包格式进入本机进行网络检测的工作了！不过，如果你的主机是作为区网的路由器， 那么建议 icmp 封包还是要通通放行才好！这是因为客户端检测网络时，常常会使用 ping 来测试到路由器的线路是否畅通之故吻！ 所以不要将路由器的 icmp 关掉，会有状况啦！

9.3.4-7 超阳春客户端防火墙设计与防火墙规则储存

经过上述的本机 iptables 语法分析后，接下来我们来想想，如果站在客户端且不提供网络服务的 Linux 本机角色时， 你应该要如何设计你的防火墙呢？老实说，你只要分析过 CentOS 默认的防火墙规则就会知道了，理论上， 应该要有的规则如下：

1. 规则归零：清除所有已经存在的规则 (iptables -F...)
2. 预设政策：除了 INPUT 这个自定义链设为 DROP 外，其他为预设 ACCEPT；
3. 信任本机：由于 lo 对本机来说是相当重要的，因此 lo 必须设定为信任装置；
4. 回应封包：让本机主动向外要求而响应的封包可以进入本机 (ESTABLISHED,RELATED)
5. 信任用户 ：这是非必要的，如果你想要让区网的来源可用你的主机资源时

这就是最最阳春的防火墙，你可以透过第二步骤抵挡所有远程的来源封包，而透过第四步骤让你要求的远程主机响应封包可以进入， 加上让本机的 lo 这个内部循环装置可以放行，嘿嘿！一部 client 专用的防火墙规则就 OK 了！你可以在某个 script 上面这样做即可：

```
[root@www ~]# vim bin/firewall.sh
#!/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin; export PATH

# 1\. 清除规则
iptables -F
iptables -X
iptables -Z

# 2\. 设定政策
iptables -P   INPUT DROP
iptables -P  OUTPUT ACCEPT
iptables -P FORWARD ACCEPT

# 3~5\. 制订各项规则
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -i eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT
#iptables -A INPUT -i eth0 -s 192.168.1.0/24 -j ACCEPT

# 6\. 写入防火墙规则配置文件
/etc/init.d/iptables save

[root@www ~]# sh bin/firewall.sh
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]
```

其实防火墙也是一个服务，你可以透过『chkconfig --list iptables』去察看就知道了。 因此，你这次修改的各种设定想要在下次开机还保存，那就得要进行『 /etc/init.d/iptables save 』这个指令加参数。 因此，鸟哥现在都是将储存的动作写入这个 firewall.sh 脚本中，比较单纯些啰！现在，你的 Linux 主机已经有相当的保护了， 只是如果想要作为服务器，或者是作为路由器，那就得要自行加上某些自定义的规则啰。

Tips: 老实说，如果你对 Linux 够熟悉的话，直接去修改 /etc/sysconfig/iptables 然后将 iptables 这个服务 restart， 那你的防火墙规则就是会在开机后持续存在啰！不过，鸟哥个人还是喜欢写 scripts 就是了。



制订好规则后当然就是要测试啰！那么如何测试呢？

1. 先由主机向外面主动联机试看看；
2. 再由私有网域内的 PC 向外面主动联机试看看；
3. 最后，由 Internet 上面的主机，主动联机到你的 Linux 主机试看看；

一步一步作下来，看看问题出在哪里，然后多多的去改进、改良！基本上，网络上目前很多的资料可以提供你不错的参考了！ 这一篇的设定写的是很简单，大部分都还在介绍阶段而已！希望对大家有帮助！ 鸟哥在参考数据(注2)当中列出几个有用的防火墙网页，希望大家有

空真的要多多的去看看！会很有帮助的！

9.3.5 IPv4 的核心管理功能： /proc/sys/net/ipv4/*

除了 iptables 这个防火墙软件之外，其实咱们 Linux kernel 2.6 提供很多核心预设的攻击抵挡机制喔！ 由于是核心的网络功能，所以相关的设定数据都是放置在 /proc/sys/net/ipv4/ 这个目录当中。 至于该目录下各个档案的详细资料，可以参考核心的说明文件 (你得要先安装 kernel-doc 软件)：

- /usr/share/doc/kernel-doc-2.6.32/Documentation/networking/ip-sysctl.txt

鸟哥这里也放一份备份：

- http:/linux.vbird.org/linux_server/0250simple_firewall/ip-sysctl.txt

有兴趣的话应该要自行去查一查比较好的喔！我们底下就拿几个简单的档案来作说明吧！

- /proc/sys/net/ipv4/tcp_syncookies

  我们在前一章谈到所谓的阻断式服务 (DoS) 攻击法当中的一种方式，就是利用 TCP 封包的 SYN 三向交握原理所达成的， 这种方式称为 SYN Flooding 。那如何预防这种方式的攻击呢？我们可以启用核心的 SYN Cookie 模块啊！ 这个 SYN Cookie 模块可以在系统用来启动随机联机的埠口 (1024:65535) 即将用完时自动启动。

  当启动 SYN Cookie 时，主机在发送 SYN/ACK 确认封包前，会要求 Client 端在短时间内回复一个序号，这个序号包含许多原本 SYN 封包内的信息，包括 IP、port 等。若 Client 端可以回复正确的序号，那么主机就确定该封包为可信的，因此会发送 SYN/ACK 封包，否则就不理会此一封包。

  透过此一机制可以大大的降低无效的 SYN 等待埠口，而避免 SYN Flooding 的 DoS 攻击说！ 那么如何启动这个模块呢？很简单，这样做即可：

  ```
  [root@www ~]# echo "1" &gt; /proc/sys/net/ipv4/tcp_syncookies
  ```

  但是这个设定值由于违反 TCP 的三向交握 (因为主机在发送 SYN/ACK 之前需要先等待 client 的序号响应)， 所以可能会造成某些服务的延迟现象，例如 SMTP (mail server)。不过总的来说，这个设定值还是不错用的！ 只是不适合用在负载已经很高的服务器内喔！ 因为负载太高的主机有时会让核心误判遭受 SYN Flooding 的攻击呢。

  如果是为了系统的 TCP 封包联机优化，则可以参考 tcp_max_syn_backlog, tcp_synack_retries, tcp_abort_on_overflow 这几个设定值的意义。

- /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

阻断式服务常见的是 SYN Flooding ，不过，我们知道系统其实可以接受使用 ping 的响应， 而 ping 的封包数据量是可以给很大的！想象一个状况， 如果有个搞破坏的人使用 1000 台主机传送 ping 给你的主机，而且每个 ping 都高达数百 K bytes时， 你的网络带宽会怎样？要嘛就是带宽被吃光，要嘛可能系统会当机！ 这种方式分别被称为 ping flooding (不断发 ping) 及 ping of death (发送大的 ping 封包)。

那如何避免呢？取消 ICMP 类型 8 的 ICMP 封包回应就是了。我们可以透过防火墙来抵挡， 这也是比较建议的方式。当然也可以让核心自动取消 ping 的响应。不过你必须要了解， 某些局域网络内常见的服务 (例如动态 IP 分配 DHCP 协议) 会使用 ping 的方式来侦测是否有重复的 IP ，所以你最好不要取消所有的 ping 响应比较好。

核心取消 ping 回应的设定值有两个，分别是：/proc/sys/net/ipv4 内的 icmp_echo_ignore_broadcasts (仅有 ping broadcast 地址时才取消 ping 的回应) 及 icmp_echo_ignore_all (全部的 ping 都不回应)。鸟哥建议设定 icmp_echo_ignore_broadcasts 就好了。 你可以这么做：

```
[root@www ~]# echo "1" &gt;  \
&gt; /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
```

- /proc/sys/net/ipv4/conf/网络接口/*

  咱们的核心还可以针对不同的网络接口进行不一样的参数设定喔！网络接口的相关设定放置在 /proc/sys/net/ipv4/conf/ 当中，每个接口都以接口代号做为其代表，例如 eth0 接口的相关设定数据在 /proc/sys/net/ipv4/conf/eth0/ 内。那么网络接口的设定数据有哪些比较需要注意的呢？ 大概有底下这几个：

  - rp_filter：称为逆向路径过滤 (Reverse Path Filtering)， 可以藉由分析网络接口的路由信息配合封包的来源地址，来分析该封包是否为合理。举例来说，你有两张网卡，eth0 为 192.168.1.10/24 ，eth1 为 public IP 。那么当有一个封包自称来自 eth1 ，但是其 IP 来源为 192.168.1.200 ， 那这个封包就不合理，应予以丢弃。这个设定值建议可以启动的。

  - log_martians：这个设定数据可以用来启动记录不合法的 IP 来源， 举例来说，包括来源为 0.0.0.0、127.x.x.x、及 Class E 的 IP 来源，因为这些来源的 IP 不应该应用于 Internet 啊。 记录的数据默认放置到核心放置的登录档 /var/log/messages。

  - accept_source_route：或许某些路由器会启动这个设定值， 不过目前的设备很少使用到这种来源路由，你可以取消这个设定值。

  - accept_redirects：当你在同一个实体网域内架设一部路由器， 但这个实体网域有两个 IP 网域，例如 192.168.0.0/24, 192.168.1.0/24。此时你的 192.168.0.100 想要向 192.168.1.100 传送讯息时，路由器可能会传送一个 ICMP redirect 封包告知 192.168.0.100 直接传送数据给 192.168.1.100 即可，而不需透过路由器。因为 192.168.0.100 与 192.168.1.100确实是在同一个实体线路上 (两者可以直接互通)，

所以路由器会告知来源 IP 使用最短路径去传递数据。但那两部主机在不同的 IP 段，却是无法实际传递讯息的！这个设定也可能会产生一些轻微的安全风险，所以建议关闭他。

- send_redirects：与上一个类似，只是此值为发送一个 ICMP redirect 封包。 同样建议关闭。(事实上，鸟哥就曾经为了这个 ICMP redirect 的问题伤脑筋！其实关闭 redirect 的这两个项目即可啊！) 虽然你可以使用『 echo "1" > /proc/sys/net/ipv4/conf/???/rp_filter 』之类的方法来启动这个项目，不过， 鸟哥比较建议修改系统设定值，那就是 /etc/sysctl.conf 这个档案！假设我们仅有 eth0 这个以太接口，而且上述的功能要通通启动， 那你可以这样做：

```
[root@www ~]# vim /etc/sysctl.conf
# Adding by VBird 2011/01/28
net.ipv4.tcp_syncookies = 1
net.ipv4.icmp_echo_ignore_broadcasts = 1
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.default.rp_filter = 1
net.ipv4.conf.eth0.rp_filter = 1
net.ipv4.conf.lo.rp_filter = 1
....(以下省略)....

[root@www ~]# sysctl -p
```

9.4 单机防火墙的一个实例

介绍了这么多的防火墙语法与相关的注意事项后，终于要来架设防火墙了。鸟哥还是比较偏好使用脚本来撰写防火墙， 然后透过最终的 /etc/init.d/iptables save 来将结果储存到 /etc/sysconfig/iptables 去！ 而且此一特色还可以用在呼叫其他的 scripts ，可以让防火墙规则具有较为灵活的使用方式。 好了，那就来谈谈如何设定咱们的防火墙规则吧！

9.4.1 规则草拟

鸟哥底下介绍的这个防火墙，其实可以用来作为路由器上的防火墙，也可以用来作为本机的防火墙。 假设硬件联机如同下图所示， Linux 主机本身也是内部 LAN 的路由器！亦即是一个简单的 IP 分享器的功能啦！依据第三章的图 3.2-1 假设鸟哥网络接口有底下这些：

- 外部网络使用 eth0 (如果是拨接，有可能是 ppp0，请针对你的环境来设定)；
- 内部网络使用 eth1 ，且内部使用 192.168.100.0/24 这个 Class ；
- 主机默认开放的服务有 WWW, SSH, https 等等；

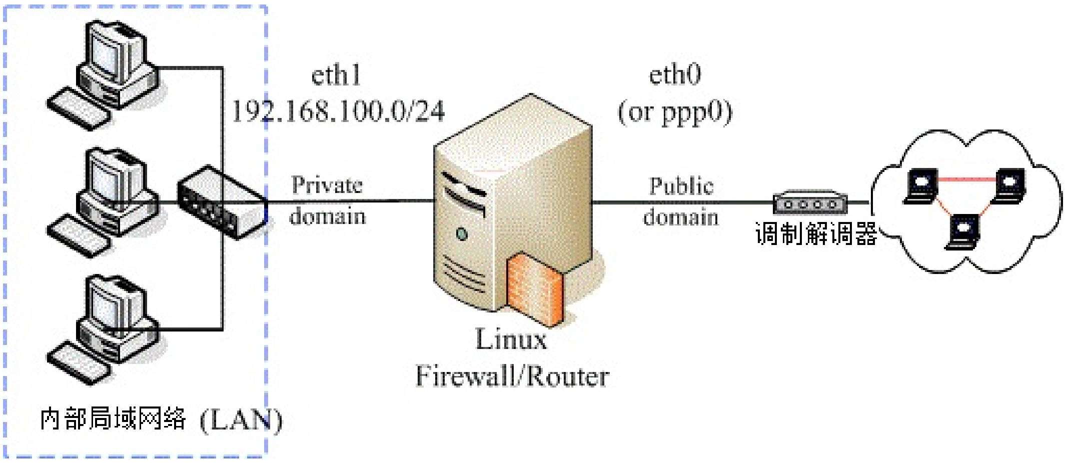


图 9.4-1、一个局域网络的路由器架构示意图

由于希望将信任网域 (LAN) 与不信任网域 (Internet) 整个分开的完整一点， 所以希望你可以在 Linux 上面安装两块以上的实体网卡，将两块网卡接在不同的网域，这样可以避免很多问题。 至于最重要的防火墙政策是：『关闭所有的联机，仅开放特定的服务』模式。 而且假设内部使用者已经受过良好的训练，因此在 filter table 的三条链个预设政策是：

- INPUT 为 DROP
- OUTPUT 及 FORWARD 为 ACCEPT

鸟哥底下预计提供的防火墙流程是这样的：

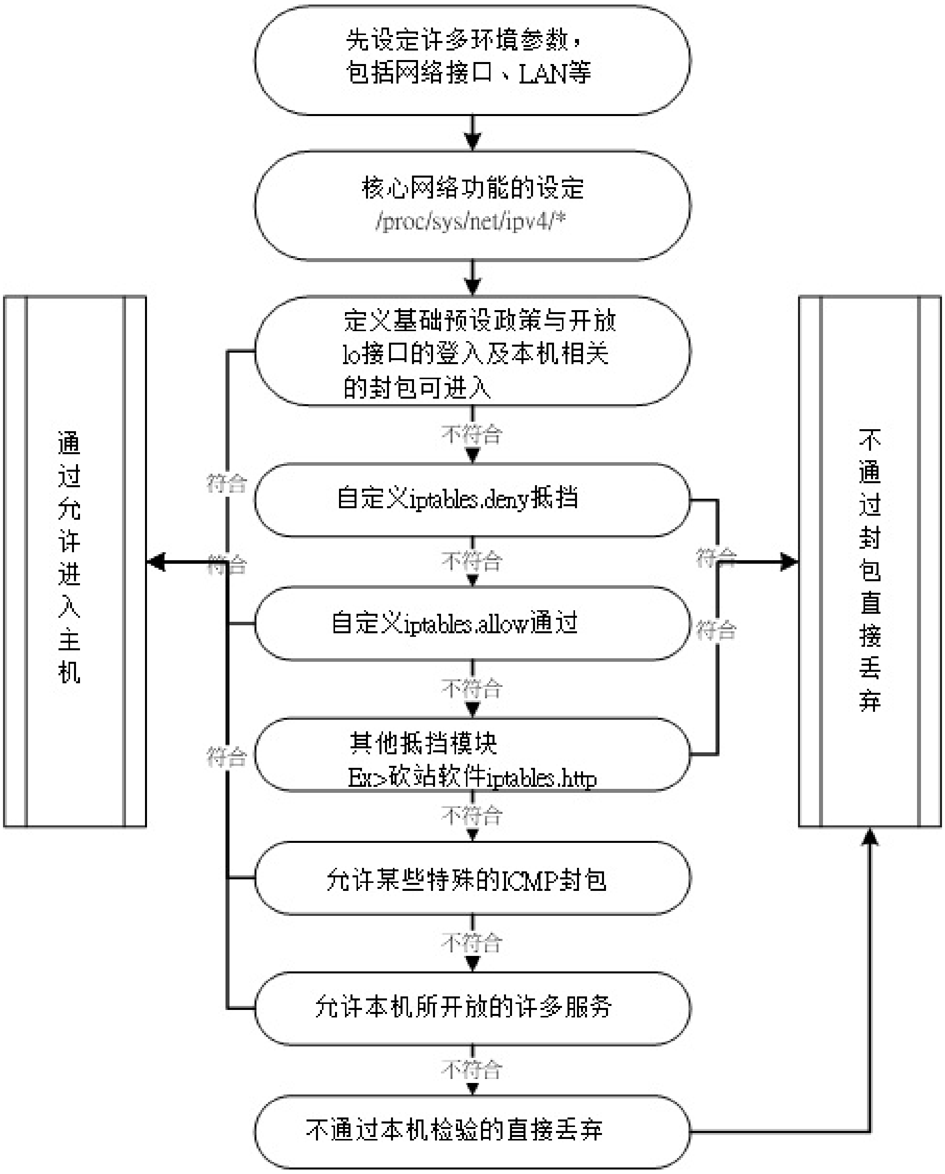


图 9.4-2、本机的防火墙规则流程示意图

原则上，内部 LAN 主机与主机本身的开放度很高，因为 Output 与 Forward 是完全开放不理的！对于小家庭的主机是可以接受的，因为我们内部的计算机数量不多，而且人员都是熟悉的， 所以不需要特别加以控管！但是：『在大企业的内部，这样的规划是很不合格的， 因为你不能保证内部所有的人都可以按照你的规定来使用 Network ！』也就是说『家贼难防』呀！ 因此，那样的环境连 Output 与 Forward 都需要特别加以管理才行！

9.4.2 实际设定

事实上，我们在设定防火墙的时候，不太可能会一个一个指令的输入，通常是利用 shell scripts 来帮我们达成这样的功能呐！底下是利用上面的流程图所规划出来的防火墙脚本，你可以参考看看， 但是你需要将环境修改成适合你自己的环境才行喔！此外，为了未来修改维护的方便，鸟哥将整个 script 拆成三部分，分别是：

- iptables.rule：设定最基本的规则，包括清除防火墙规则、加载模块、设定服务可接受等；
- iptables.deny：设定抵挡某些恶意主机的进入；
- iptables.allow：设定允许某些自定义的后门来源主机！

鸟哥个人习惯是将这个脚本放置到 /usr/local/virus/iptables 目录下，你也可以自行放置到自己习惯的位置去。 那底下就来瞧瞧这支脚本是怎么写的吧！

```
[root@www ~]# mkdir -p /usr/local/virus/iptables
[root@www ~]# cd /usr/local/virus/iptables
[root@www iptables]# vim iptables.rule
#!/bin/bash

# 请先输入您的相关参数，不要输入错误了！
  EXTIF="eth0"             # 这个是可以连上 Public IP 的网络接口
  INIF="eth1"              # 内部 LAN 的连接接口；若无则写成 INIF=""
  INNET="192.168.100.0/24" # 若无内部网域接口，请填写成 INNET=""
  export EXTIF INIF INNET

# 第一部份，针对本机的防火墙设定！##########################################
# 1\. 先设定好核心的网络功能：
  echo "1" &gt; /proc/sys/net/ipv4/tcp_syncookies
  echo "1" &gt; /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
  for i in /proc/sys/net/ipv4/conf/*/{rp_filter,log_martians}; do
        echo "1" &gt; $i
  done
  for i in /proc/sys/net/ipv4/conf/*/{accept_source_route,accept_redirects,\
send_redirects}; do
        echo "0" &gt; $i
  done

# 2\. 清除规则、设定默认政策及开放 lo 与相关的设定值
  PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/sbin:/usr/local/bin; export PATH
  iptables -F
  iptables -X
  iptables -Z
  iptables -P INPUT   DROP
  iptables -P OUTPUT  ACCEPT
  iptables -P FORWARD ACCEPT
  iptables -A INPUT -i lo -j ACCEPT
  iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

# 3\. 启动额外的防火墙 script 模块
  if [ -f /usr/local/virus/iptables/iptables.deny ]; then
        sh /usr/local/virus/iptables/iptables.deny
  fi
  if [ -f /usr/local/virus/iptables/iptables.allow ]; then
        sh /usr/local/virus/iptables/iptables.allow
  fi
  if [ -f /usr/local/virus/httpd-err/iptables.http ]; then
        sh /usr/local/virus/httpd-err/iptables.http
  fi

# 4\. 允许某些类型的 ICMP 封包进入
  AICMP="0 3 3/4 4 11 12 14 16 18"
  for tyicmp in $AICMP
  do
    iptables -A INPUT -i $EXTIF -p icmp --icmp-type $tyicmp -j ACCEPT
  done

# 5\. 允许某些服务的进入，请依照你自己的环境开启
# iptables -A INPUT -p TCP -i $EXTIF --dport  21 --sport 1024:65534 -j ACCEPT # FTP
# iptables -A INPUT -p TCP -i $EXTIF --dport  22 --sport 1024:65534 -j ACCEPT # SSH
# iptables -A INPUT -p TCP -i $EXTIF --dport  25 --sport 1024:65534 -j ACCEPT # SMTP
# iptables -A INPUT -p UDP -i $EXTIF --dport  53 --sport 1024:65534 -j ACCEPT # DNS
# iptables -A INPUT -p TCP -i $EXTIF --dport  53 --sport 1024:65534 -j ACCEPT # DNS
```

```
# iptables -A INPUT -p TCP -i $EXTIF --dport  80 --sport 1024:65534 -j ACCEPT # WWW
# iptables -A INPUT -p TCP -i $EXTIF --dport 110 --sport 1024:65534 -j ACCEPT # POP3
# iptables -A INPUT -p TCP -i $EXTIF --dport 443 --sport 1024:65534 -j ACCEPT # HTTPS

# 第二部份，针对后端主机的防火墙设定！###############################
# 1\. 先加载一些有用的模块
  modules="ip_tables iptable_nat ip_nat_ftp ip_nat_irc ip_conntrack
ip_conntrack_ftp ip_conntrack_irc"
  for mod in $modules
  do
      testmod=`lsmod &#124; grep "^${mod} " &#124; awk '{print $1}'`
      if [ "$testmod" == "" ]; then
            modprobe $mod
      fi
  done

# 2\. 清除 NAT table 的规则吧！
  iptables -F -t nat
  iptables -X -t nat
  iptables -Z -t nat
  iptables -t nat -P PREROUTING  ACCEPT
  iptables -t nat -P POSTROUTING ACCEPT
  iptables -t nat -P OUTPUT      ACCEPT

# 3\. 若有内部接口的存在 (双网卡) 开放成为路由器，且为 IP 分享器！
  if [ "$INIF" != "" ]; then
    iptables -A INPUT -i $INIF -j ACCEPT
    echo "1" &gt; /proc/sys/net/ipv4/ip_forward
    if [ "$INNET" != "" ]; then
        for innet in $INNET
        do
            iptables -t nat -A POSTROUTING -s $innet -o $EXTIF -j MASQUERADE
        done
    fi
  fi
 # 如果你的 MSN 一直无法联机，或者是某些网站 OK 某些网站不 OK，
  # 可能是 MTU 的问题，那你可以将底下这一行给他取消批注来启动 MTU 限制范围
  # iptables -A FORWARD -p tcp -m tcp --tcp-flags SYN,RST SYN -m tcpmss \
  #          --mss 1400:1536 -j TCPMSS --clamp-mss-to-pmtu

# 4\. NAT 服务器后端的 LAN 内对外之服务器设定
# iptables -t nat -A PREROUTING -p tcp -i $EXTIF --dport 80 \
#          -j DNAT --to-destination 192.168.1.210:80 # WWW

# 5\. 特殊的功能，包括 Windows 远程桌面所产生的规则，假设桌面主机为 1.2.3.4
# iptables -t nat -A PREROUTING -p tcp -s 1.2.3.4  --dport 6000 \
#          -j DNAT --to-destination 192.168.100.10
# iptables -t nat -A PREROUTING -p tcp -s 1.2.3.4  --sport 3389 \
#          -j DNAT --to-destination 192.168.100.20

# 6\. 最终将这些功能储存下来吧！
  /etc/init.d/iptables save
```

特别留意上面程序代码的特殊字体部分，基本上，你只要修改一下最上方的接口部分， 应该就能够运作这个防火墙了。不过因为每个人的环境都不相同， 因此你在设定完成后，依旧需要测试一下才行喔！不然，出了问题不要怪我啊！.... 再来看一下关于 iptables.allow 的内容是如何？假如我要让一个 140.116.44.0/24 这个网域的所有主机来源可以进入我的主机的话，那么这个档案的内容可以写成这样：

```
[root@www iptables]# vim iptables.allow
#!/bin/bash
# 底下则填写你允许进入本机的其他网域或主机啊！
  iptables -A INPUT -i $EXTIF -s 140.116.44.0/24 -j ACCEPT

# 底下则是关于抵挡的档案设定法！
[root@www iptables]# vim iptables.deny
#!/bin/bash
# 底下填写的是『你要抵挡的那个咚咚！』
  iptables -A INPUT -i $EXTIF -s 140.116.44.254 -j DROP

[root@www iptables]# chmod 700 iptables.*
```

将这三个档案的权限设定为 700 且只属于 root 的权限后，就能够直接执行 iptables.rule 啰！不过要注意的是，在上面的案例当中，鸟哥预设将所有的服务的通道都是关闭的！ 所以你必须要到本机防火墙的第 5 步骤处将一些批注符号 (#) 解开才行。 同样的，如果有其他更多的 port 想要开启时，一样需要增加额外的规则才行喔！

不过，还是如同前面我们所说的，这个 firewall 仅能提供基本的安全防护，其他的相关问题还需要再测试测试呢！ 此外，如果你希望一开机就自动执行这个 script 的话，请将这个档案的完整档名写入 /etc/rc.d/rc.local 当中，有点像底下这样：

```
[root@www ~]# vim /etc/rc.d/rc.local
....(其他省略)....
# 1\. Firewall
/usr/local/virus/iptables/iptables.rule
```

事实上，这个脚本的最底下已经加入写入防火墙默认规则文件的功能，所以你只要执行一次，就拥有最正确的规则了！ 上述的 rc.local 仅是预防万一而已。^_^！上述三个档案请你不要在 Windows 系统上面编辑后才传送到 Linux 上运作，因为 Windows 系统的断行字符问题，将可能导致该档案无法执行。建议你直接到底下去下载，传送到 Linux 后可以利用 dos2unix 指令去转换断行字符！就不会有问题！

- http://linux.vbird.org/download/index.php?action=detail&fileid=43

这就是一个最简单、阳春的防火墙。同时，这个防火墙还可以具有最阳春的 IP 分享器的功能呢！ 也就是在 iptables.rule 这个档案当中的第二部分了。 这部分我们在下一节会再继续介绍的。

9.5 NAT 服务器的设定

呼呼！终于来到这个地方了！我们准备要架设一个路由器的延伸服务器，就称之为 NAT 服务器。 NAT 是什么呢？简单的说，你可以称他为内部 LAN 主机的『 IP 分享器』啦！

NAT 的全名是 Network Address Translation，字面上的意思是『网络地址的转换』。由字面上的意思我们来想一想， TCP/IP 的网络封包不是有 IP 地址吗？那 IP 地址不是有来源与目的吗？我们的 iptables 指令就能够修改 IP 封包的表头数据， 嘿嘿！连目标或来源的 IP 地址都可以修改呢！甚至连 TCP 封包表头的 port number 也能修改！真是有趣！

NAT 服务器的功能可以达到类似图 9.1-2所介绍的类似 IP 分享的功能之外， 还可以达到类似图 9.1-4所介绍的 DMZ (非军事区) 的功能！这完全取决于我们的 NAT 是修改： (1)来源 IP 还是 (2)目标 IP ！底下我们就来聊一聊吧！ ^_^

9.5.1 什么是 NAT？ SNAT？ DNAT？

在谈到 NAT 的实际运作之前，让我们再来看一下比较简单的封包透过 iptables 而传送到后端主机的表格与链流程(请往前参考图 9.3-4)。 当网络布线如图 9.1-2的架构，若内部 LAN 有任何一部主机想要传送封包出去时， 那么这个封包要如何透过 Linux 主机而传送出去？他是这样的：

1. 先经过 NAT table 的 PREROUTING 链；
2. 经由路由判断确定这个封包是要进入本机与否，若不进入本机，则下一步；
3. 再经过 Filter table 的 FORWARD 链；
4. 通过 NAT table 的 POSTROUTING 链，最后传送出去。

NAT 服务器的重点就在于上面流程的第 1,4 步骤，也就是 NAT table 的两条重要的链：PREROUTING 与 POSTROUTING。 那这两条链有什么重要的功能呢？重点在于修改 IP 嘛！但是这两条链修改的 IP 是不一样的！ POSTROUTING 在修改来源 IP ，PREROUTING 则在修改目标 IP 。 由于修改的 IP 不一样，所以就称为来源 NAT (Source NAT, SNAT) 及目标 NAT (Destination NAT, DNAT)。我们先来谈一谈 IP 分享器功能的 SNAT 吧！

- 来源 NAT, SNAT：修改封包表头的『来源』项目

  你应该有听说过 IP 分享器这个玩意儿，他可以让你家庭里的好几部主机同时透过一条 ADSL 网络联机到 Internet 上面， 例如图 9.1-2联机的方式来说，那个 Linux 主机就是 IP 分享器啦！那么他是如何达到 IP 分享的功能？就是透过 NAT 表格的 POSTROUTING 来处理的。假设你的网络布线如图 9.1-2所示， 那么 NAT 服务器是如何处理这个封包的呢？

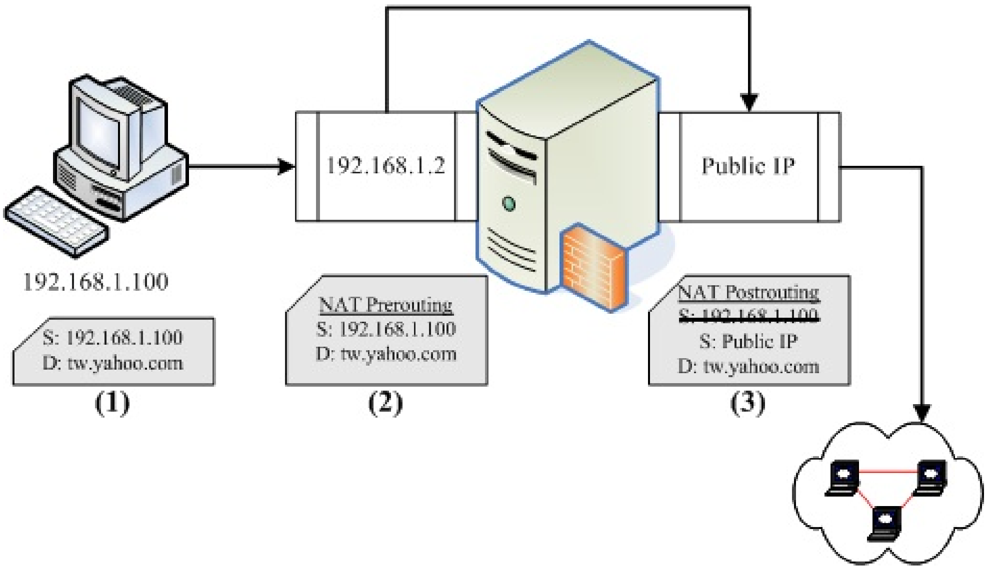


图 9.5-1、

SNAT 封包传送出去的示意图

如上图所示，在客户端 192.168.1.100 这部主机要联机到 http://tw.yahoo.com 去时，他的封包表头会如何变化？

1. 客户端所发出的封包表头中，来源会是 192.168.1.100 ，然后传送到 NAT 这部主机；
2. NAT 这部主机的内部接口 (192.168.1.2) 接收到这个封包后，会主动分析表头数据，因为表头数据显示目的并非 Linux 本机，所以开始经过路由， 将此封包转到可以连接到 Internet 的 Public IP 处；
3. 由于 private IP 与 public IP 不能互通，所以 Linux 主机透过 iptables 的 NAT table 内的 Postrouting 链将封包表头的来源伪装成为 Linux 的 Public IP ，并且将两个不同来源 (192.168.1.100 及 public IP) 的封包对应写入暂存内存当中， 然后将此封包传送出去了； 此时 Internet 上面看到这个封包时，都只会知道这个封包来自那个 Public IP 而不知道其实是来自内部啦。 好了，那么如果 Internet 回传封包呢？又会怎么作？

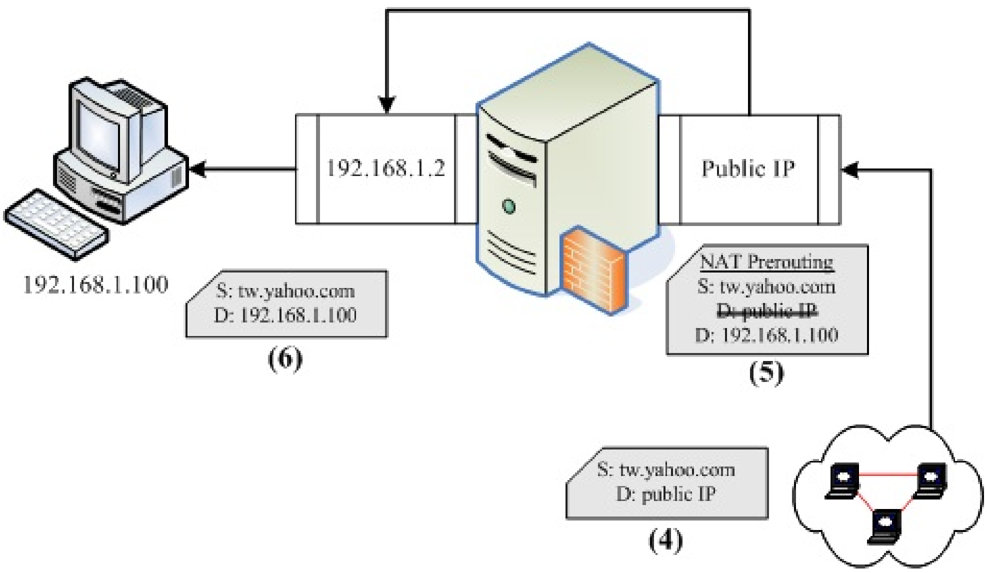


图 9.5-2、

SNAT 封包接收的示意图

1. 在 Internet 上面的主机接到这个封包时，会将响应数据传送给那个 Public IP 的主机；
2. 当 Linux NAT 服务器收到来自 Internet 的回应封包后，会分析该封包的序号，并比对刚刚记录到内存当中的数据， 由于发现该封包为后端主机之前传送出去的，因此在 NAT Prerouting 链中，会将目标 IP 修改成为后端主机，亦即那部 192.168.1.100，然后发现目标已经不是本机 (public IP)， 所以开始透过路由分析封包流向；
3. 封包会传送到 192.168.1.2 这个内部接口，然后再传送到最终目标 192.168.1.100 机器上去！ 经过这个流程，你就可以发现到，所有内部 LAN 的主机都可以透过这部 NAT 服务器联机出去， 而大家在 Internet 上面看到的都是同一个 IP (就是 NAT 那部主机的 public IP 啦！)， 所以，如果内部 LAN 主机没有连上不明网站的话，那么内部主机其实是具有一定程度的安全性的啦！ 因为 Internet 上的其他主机没有办法主动攻击你的 LAN 内的 PC 嘛！所以我们才会说， NAT 最简单的功能就是类似 IP 分享器啦！那也是 SNAT 的一种。

Tips: NAT 服务器与路由器有啥不同？基本上，NAT 服务器一定是路由器，不过， NAT 服务器由于会修改 IP 表头数据， 因此与单纯转递封包的路由器不同。最常见的 IP 分享器就是一个路由器，但是这个 IP 分享器一定会有一个 Public IP 与一个 Private IP，让 LAN 内的 Private IP 可以透过 IP 分享器的 Public IP 传送出去喔！ 至于路由器通常两边都是 Public IP 或同时为 Private IP。



- 目标 NAT, DNAT：修改封包表头的『目标』项目

SNAT 主要是应付内部 LAN 连接到 Internet 的使用方式，至于 DNAT 则主要用在内部主机想要架设可以让 Internet 存取的服务器啦！ 就有点类似图 9.1-4的 DMZ 内的服务器啊！底下也先来谈一谈 DNAT 的运作吧！

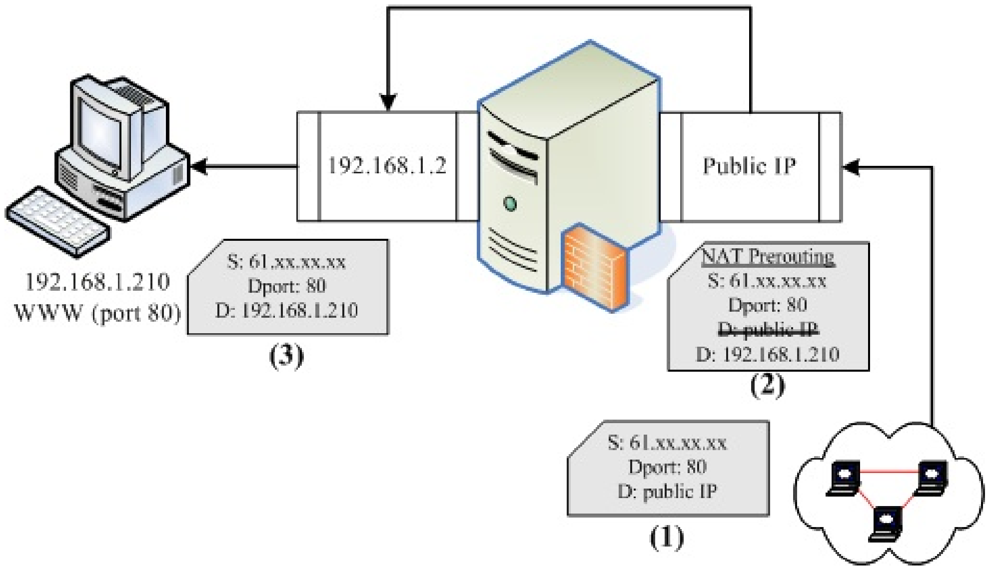


图 9.5-3、DNAT 的封包传送示意图

如上图所示，假设我的内部主机 192.168.1.210 启动了 WWW 服务，这个服务的 port 开启在 port 80 ， 那么 Internet 上面的主机 (61.xx.xx.xx) 要如何连接到我的内部服务器呢？当然啦， 还是得要透过 Linux NAT 服务器嘛！所以这部 Internet 上面的机器必须要连接到我们的 NAT 的 public IP 才行。

1. 外部主机想要连接到目的端的 WWW 服务，则必须要连接到我们的 NAT 服务器上头；
2. 我们的 NAT 服务器已经设定好要分析出 port 80 的封包，所以当 NAT 服务器接到这个封包后， 会将目标 IP 由 public IP 改成 192.168.1.210 ，且将该封包相关信息记录下来，等待内部服务器的响应；
3. 上述的封包在经过路由后，来到 private 接口处，然后透过内部的 LAN 传送到 192.168.1.210 上头！
4. 192.186.1.210 会响应数据给 61.xx.xx.xx ，这个回应当然会传送到 192.168.1.2 上头去；
5. 经过路由判断后，来到 NAT Postrouting 的链，然后透过刚刚第二步骤的记录，将来源 IP 由 192.168.1.210 改为 public IP 后，就可以传送出去了！ 其实整个步骤几乎就等于 SNAT 的反向传送哩！这就是 DNAT 啰！很简单吧！

9.5.2 最阳春 NAT 服务器： IP 分享功能

在 Linux 的 NAT 服务器服务当中，最常见的就是类似图 9.1-2的 IP 分享器功能了。 而由刚刚的介绍你也该知道，这个 IP 分享器的功能其实就是 SNAT 啦！作用就只是在 iptables 内的 NAT 表格当中，那个路由后的 POSTROUTING 链进行 IP 的伪装就是了。另外， 你也必须要了解，你的 NAT 服务器必须要有一个 public IP 接口，以及一个内部 LAN 连接的 private IP 界面才行。底下的范例中，鸟哥的假设是这样的：

- 外部接口使用 eth0 ，这个接口具有 public IP 喔；

- 内部接口使用 eth1 ，假设这个 IP 为 192.168.100.254 ；

记住！当你利用前面几章谈到的数据来设定你的网络参数后，务必要进行路由的检测， 因为在 NAT 服务器的设定方面，最容易出错的地方就是路由了！尤其是在拨接产生 ppp0 这个对外接口的环境下， 这个问题最严重。反正你要记得：『如果你的 public IP 取得的方式是拨接或 cable modem 时，你的配置文件 /etc/sysconfig/network, ifcfg-eth0, ifcfg-eth1 等档案，千万不要设定 GATEWAY 啦！』否则就会出现两个 default gateway ，反而会造成问题。

如果你刚刚已经下载了 iptables.rule ，那么该档案内已经含有 NAT 的脚本了！ 你可以看到该档案的第二部份关于 NAT 服务器的部分，应该有看到底下这几行：

```
iptables -A INPUT -i $INIF -j ACCEPT
# 这一行为非必要的，主要的目的是让内网 LAN 能够完全的使用 NAT 服务器资源。
# 其中 $INIF 在本例中为 eth1 接口

echo "1" &gt; /proc/sys/net/ipv4/ip_forward
# 上头这一行则是在让你的 Linux 具有 router 的能力

iptables -t nat -A POSTROUTING -s $innet -o $EXTIF -j MASQUERADE
# 这一行最关键！就是加入 nat table 封包伪装！本例中 $innet 是 192.168.100.0/24
# 而 $EXTIF 则是对外界面，本例中为 eth0
```

重点在那个『 MASQUERADE 』！这个设定值就是『 IP 伪装成为封包出去 (-o) 的那块装置上的 IP 』！以上面的例子来说，就是 $EXTIF ，也就是 eth0 啦！ 所以封包来源只要来自 $innet (也就是内部 LAN 的其他主机) ，只要该封包可透过 eth0 传送出去， 那就会自动的修改 IP 的来源表头成为 eth0 的 public IP 啦！就这么简单！ 你只要将 iptables.rule 下载后，并设定好你的内、外网络接口， 执行 iptables.rule 后，你的 Linux 就拥有主机防火墙以及 NAT 服务器的功能了！

例题：如同上面所述的案例，那么你的 LAN 内的其他 PC 应该要如何设定相关的网络参数？答：答案其实很简单啊，将 NAT 服务器作为 PC 的 GATEWAY 即可！只要记得底下的参数值：

- NETWORK 为 192.168.100.0
- NETMASK 为 255.255.255.0
- BROADCAST 为 192.168.100.255
- IP 可以设定 192.168.100.1 ~ 192.168.100.254 间，不可重复！
- 通讯闸 (Gateway) 需要设定为 192.168.100.254 (NAT 服务器的 Private IP)
- DNS (/etc/resolv.conf) 需设定为 168.95.1.1 (Hinet) 或 139.175.10.20 (Seed Net)，这个请依你的 ISP 而定；

事实上，除了 IP 伪装 (MASQUERADE) 之外，我们还可以直接指定修改 IP 封包表头的来源 IP 呢！ 举例来说，如下面这个例子：

例题：假设对外的 IP 固定为 192.168.1.100 ，若不想使用伪装，该如何处理？答：

```
iptables -t nat -A POSTROUTING -o eth0 -j SNAT \
         --to-source 192.168.1.100
```

例题：假设你的 NAT 服务器对外 IP 有好几个，那你想要轮流使用不同的 IP 时，又该如何设定？举例来说，你的 IP 范围为 192.168.1.210~192.168.1.220答：

```
iptables -t nat -A POSTROUTING -o eth0 -j SNAT \
         --to-source 192.168.1.210-192.168.1.220
```

这样也可以修改网络封包的来源 IP 资料喔！不过，除非你使用的是固定 IP ，且有多个 IP 可以对外联机，否则一般使用 IP 伪装即可，不需要使用到这个 SNAT 啦！当然，你也可能有自己的独特的环境啦！ ^_^

9.5.3 iptables 的额外核心模块功能

如果你刚刚在 iptables.rule 内的第二部分有仔细看的话， 那有没有觉得很奇怪，为何我们需要加载一些有用的模块？举例来说， ip*nat_ftp* 及 *ip_nat_irc* ？ 这是因为很多通讯协议使用的封包传输比较特殊，尤其是 *FTP* 文件传输使用到两个 *port* 来处理数据！ 这个部分我们会在 *FTP* 章节再来详谈，在这里你要先知道，我们的 *iptables* 提供很多好用的模块， 这些模块可以辅助封包的过滤用途，让我们可以节省很多 *iptables* 的规则拟定，好棒的呐！ ^^

9.5.4 在防火墙后端之网络服务器 DNAT 设定

既然可以做 SNAT 的 IP 分享功能，我们当然可以使用 iptables 做出 DMZ 啦！ 但是再次重申，不同的服务器封包传输的方式可能有点差异，因此，建议新手不要玩这个咚咚！ 否则很容易导致某些服务无法顺利对 Internet 提供的问题。

先来谈一谈，如果我想要处理 DNAT 的功能时， iptables 要如何下达指令？ 另外，你必须要知道的是， DNAT 用到的是 nat table 的 Prerouting 链喔！不要搞错了。

例题：假设内网有部主机 IP 为 192.168.100.10 ，该主机是可对 Internet 开放的 WWW 服务器。你该如何透过 NAT 机制，将 WWW 封包传到该主机上？答：假设 public IP 所在的接口为 eth0 ，那么你的规则就是：

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 \
     -j DNAT --to-destination 192.168.100.10:80
```

那个『 -j DNAT --to-destination IP[:port] 』就是精髓啦！代表从 eth0 这个接口传入的，且想要使用 port 80 的服务时， 将该封包重新传导到 192.168.100.10:80 的 IP 及 port 上面！可以同时修改 IP 与 port 呢！真方便。 其他还有一些较进阶的 iptables 使用方式，如下所示：

```
-j REDIRECT --to-ports &lt;port number&gt;
# 这个也挺常见的，基本上，就是进行本机上面 port 的转换就是了！
# 不过，特别留意的是，这个动作仅能够在 nat table 的 PREROUTING 以及
# OUTPUT 链上面实行而已喔！

范例：将要求与 80 联机的封包转递到 8080 这个 port
[root@www ~]# iptables -t nat -A PREROUTING -p tcp  --dport 80 \
&gt; -j REDIRECT --to-ports 8080
# 这玩意最容易在你使用了非正规的 port 来进行某些 well known 的协议，
# 例如使用 8080 这个 port 来启动 WWW ，但是别人都以 port 80 来联机，
# 所以，你就可以使用上面的方式来将对方对你主机的联机传递到 8080 啰！
```

至于更多的用途，那就有待你自己的发掘啰！^_^

9.6 重点回顾

- 要拥有一部安全的主机，必须要有良好的主机权限设定；随时的更新套件；定期的重要数据备份；完善的员工教育训练。 仅有防火墙是不足够的；
- 防火墙最大的功能就是帮助你『限制某些服务的存取来源』，可以管制来源与目标的 IP ；
- 防火墙依据封包抵挡的阶层，可以分为 Proxy 以及 IP Filter (封包过滤) 两种类型；
- 在防火墙内，但不在 LAN 内的服务器所在网域，通常被称为 DMZ (非军事区)，如图 9.1-4所示；
- 封包过滤机制的防火墙，通常至少可以分析 IP, port, flag (如 TCP 封包的 SYN), MAC 等等；
- 防火墙对于病毒的抵挡并不敏感；
- 防火墙对于来自内部的网络误用或滥用的抵挡性可能较不足；
- 并不是架设防火墙之后，系统就一定很安全！还是需要更新套件漏洞以及管制使用者及权限设定等；
- 核心 2.4 以后的 Linux 使用 iptables 作为防火墙的软件；
- 防火墙的订定与『规则顺序』有很大的关系；若规则顺序错误，可能会导致防火墙的失效；
- iptables 的预设 table 共有三个，分别是 filter, nat 及 mangle ，惯用者为 filter (本机) 与 nat (后端主机)。
- filter table 主要为针对本机的防火墙设定，依据封包流向又分为 INPUT, OUTPUT, FORWARD 三条链；
- nat table 主要针对防火墙的后端主机，依据封包流向又分为 PREROUTING, OUTPUT, POSTROUTING 三条链， 其中 PREROUTING 与 DNAT 有关， POSTROUTING 则与 SNAT 有关；
- iptables 的防火墙为规则比对，但所有规则都不符合时，则以预设政策 (policy) 作为封包的行为依据；
- iptables 的指令列当中，可以下达的参数相当的多，当下达 -j LOG 的参数时，则该封包的流程会被纪录到 /var/log/messages 当中；
- 防火墙可以多重设定，例如虽然已经设定了 iptables ，但是仍然可以持续设定 TCP Wrappers ，因为谁也不晓得什么时候 iptables 会有漏洞～或者是规则规划不良！

9.7 本章习题

- 为什么我架设了防火墙，我的主机还是可能中毒？防火墙不是万灵丹，他还是可能被病毒或者是木马程序所入侵的！ 此外，如果你的主机本身就已经提供了多个网络服务，则当该网络服务的软件有漏洞时， 防火墙仍然无法克服该服务的漏洞的！因此仍然需要持续的进行主机的监视与后端分析工作
- 请说明为何架设了防火墙，我的主机还是可能被入侵？入侵的依据可能是什么方法？因为防火墙仅是抵挡某些不受欢迎的封包，如果你有开放 WWW 的服务时，则要求你主机 port 80 的封包将可直接进入你的主机，万一 WWW 套件有漏洞时，那么就可能被入侵了！所以软件的更新很重要！
- 我们知道核心为 2.6 的 Linux 使用的防火墙机制为 iptables ，请问，如何知道我的 Linux 核心版本？利用 uname -r 可以查得！
- 请列出 iptables 预设的两个主要的 table ，以及各个 table 里面的 chains 与各个 chains 所代表的意义；filter 为预设的 Table，里头预设的链有：

    - INPUT：为来自外部，想要进入主机的封包；
    - OUTPUT：为来自主机，想要离开主机的封包；
    - FORWARD：为主机内部网域与外部网域的封包(不论进或者出)，但该封包不会进入主机。

  还有 nat 这个 table：

    - PREROUTING：进行路由之前的封包传送过程
    - OUTPUT：离开主机的封包传送过程；
    - POSTROUTING：已经经过路由了，然后才进行的过滤规则。
- 什么是 iptables 的预设政策 (Policy)？若我要针对 filter 的 INPUT 做成 DROP 的默认政策，指令如何下达？当封包的所有属性都不在防火墙的规则当中时，那么这个封包能否顺利的通过防火墙，则以 Policy 作为这个封包的最终动作了！ iptables -P INPUT DROP
- 假设今天我的 Linux 仅是作为 Client 之用，并没有对 Internet 进行任何服务， 那么你的防火墙规划应该如何设定比较好？既然没有对 Internet 提供任何服务，那么(1)请将所有的对外埠口先关闭吧！(2)防火墙规则当中，最重要的是 INPUT 的 Policy 一定要 DROP ，然后将『 iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT 』即可！
- 我要将来自 192.168.1.50 这个 IP 来源的封包，只要是向我的 21~23 埠口要求的封包，就将他抵挡，应该如何下达 iptables 指令？iptables -A INPUT -p tcp -s 192.168.1.50 --dport 21:23 -j DROP
- 我要将我自己主机 ping 的响应功能取消，应该如何下达 iptables 的指令？因为 ping 能否响应用的是 icmp 的 type 8 (请参考网络基础内的 ICMP 相关内容)，所以我可以这样做：iptables -I INPUT -p icmp --icmp-type 8 -j DROP
- 请说明为何这个指令是错误的？『iptables -A INPUT -p udp --syn -s 192.168.0.20 -j

DROP』？因为只有 TCP 封包才会具有 SYN 的标志， UDP 并没有 SYN 的标志啊！所以上面的指令是错误的

- DNS 的要求是必须的，那么我该如何设定我的主机可以接受要求 DNS 的响应呢？因为 DNS 的来源是 port 53 ，因此要接受来自 port 53 的封包就成为了： iptables -A INPUT -p udp --sport 53 -j ACCEPT iptables -A INPUT -p tcp --sport 53 -j ACCEPT
- 如何取消 iptables 在我的系统上面？先要清除规则后，才能够将 iptables 移除！不过，我们主要将规则清除即可！ iptables -F; iptables -X; iptables -Z iptables -t nat -F; iptables -t nat -X; iptables -t nat -Z
- 如何储存目前的防火墙机制，以及如何将上次储存下来的机制回复到目前的系统中？请利用 iptables-save 以及 iptables-restore 这两个指令，配合命令重导向即可！ 也可以使用 /etc/init.d/iptables save 来储存喔！
- 如果你的区网当中有个 PC 使用者老是连上 Internet 乱搞，你想要将他的 IP 锁住，但他总是有办法修改成其他 IP 来连外， 那你该怎么办？让他无法继续连外？可以利用封锁网络卡卡号 MAC 来处理！

9.8 参考数据与延伸阅读

- 注1： squid 官网：http://www.squid-cache.org/ 鸟哥的旧版文章：http://linux.vbird.org/linux_server/0420squid.php
- 注2： 与 iptables 相关的网站与书籍： 中文网站：
  - http://www.study-area.org/linux/servers/linux_nat.htm英文网站：
  - http://www.netfilter.org/
  - http://www.netfilter.org/documentation/HOWTO//packet-filtering-HOWTO.html
  - http://www.interhack.net/pubs/fwfaq/
  - http://www.sysresccd.org/Sysresccd-Networking-EN-Destination-port-routing其他书籍与数据：
  - Robert L. Ziegler 着，朱亮恺等译，『实战 Linux 防火墙--iptables 应用全搜录』，上奇出版社，2004。
  - 本机的核心文件：/usr/src/linux-{version}/networking/ip-sysctl.txt
  - iptables 的内建 tables 与各个 chain 的相关性：http://ebtables.sourceforge.net/br_fw_ia/bridge3b.png
  - 核心参数的相关说明：http://www.study-area.org/tips/adv-route/Adv-Routing-HOWTO-12.html
  - 使用 PPPoE 导致的 MTU 问题：http://www.akadia.com/services/pppoe_iptables.html

---

2002/08/20：第一次完成日期！ 2003/08/25：重新设计内容，改写一些指令介绍，与前一篇『认识网络安全』 分的比较完整一点！ 2006/09/06：将旧的文章移动到此处 2006/09/11：拿掉了已经在基础篇有介绍过的 认识服务之 TCP Wrappers。 2006/09/13：加入 NAT 的说明了，将旧的 NAT 主机移动到 此处 。 2006/09/15：将 iptables.rule 的连结贴上去了！之前忘记修改该档案了～ 2006/11/08：因为 PPPoE 拨接与 Ethernet 的 MTU 不同，可能在某些情况下会导致使用者无法联机，更新了 iptables.rule 了。 2010/10/27：将旧的基于 CentOS 4.x 的版本移动到此处 2011/02/08：修改了很多图示，并且将文句作个整理，大方向并没有特殊的修改！ 2011/07/22：将基于 CentOS 5.x 的版本移动到此处

第十章、申请合法的主机名

最近更新日期：2011/07/23

在读完了网络基础并且架设了个人简易的防火墙之后，总算是准备要开始来给他进入 Server 的架设了！服务器架设的步骤里面，很重要的一点是『你的主机名必须要在 Internet 上面可以被查询』才好！这是因为人类对于 IP 记忆力不佳，所以才会以主机名来取代 IP。不过，你的主机名要能够被查询到才有用啊！这个时候，一个『合法』的主机名就很重要了！那要合法的主机，就得要让 DNS 系统能够找的到你的主机啊！不过，如果我们的主机是使用拨接得到的不固定 IP 呢？又该如何申请 DNS 主机名？那就得要使用动态 DNS 的系统啰！在这个章节中，我们主要在介绍 Client 端的设定，而不是在设定 DNS 服务器喔！ ^_^

10.1 为何需要主机名

如果你已经将第二章的网络基础看完的话，应该会知道其实我们的 TCP/IP 环境只要有 IP 与正确的路由即可联机了。那么你申请主机名要干嘛？因为『没办法啊！人脑太不中用了！』举例来说，你可以背出来我们常上去查数据的 www.google.com 的 IP 吗？报告！鸟哥没办法背出来～

因为 IP 是那么难背的东西，而且，如果你的 IP 又是类似拨接的不固定的 IP 时，那还更伤脑筋呢！ 因此我们才会习惯以熟悉的英文字符串来做为主机名，然后让『这个主机名与 IP 达成对应』，那直接记忆主机名就行了，反正 IP 的查询就交给计算机主机来做即可！在这样的想法下，我们当然就需要有主机名啦！

Tips: 在这个章节当中，我们将会介绍如何申请一个合法的主机名。目前 Internet 上面使用的主机名都是透过所谓的 DNS 系统，而你想要取得一个 DNS 的主机名，就必须要『注册』，所谓的『注册』就是要钱去申请啦！ 当然也有免费提供主机名的服务啦！在这个章节当中鸟哥不会介绍如何架设一部 DNS 服务器， 而是介绍如何利用注册或免费申请的方式来达成主机名的取得。



10.1.1 主机名的由来

因为 IP 是这么难记忆的东西，因此人们就使用『名字』来对应到主机的 IP，这就是主机名的由来。 好在早期连上网络的计算机数量不多，所以在网络上的人们就想出一个简单的办法来进行主机名与 IP 的对应，那就是『在每部计算机的 /etc/hosts 里面设定好主机名与 IP 的对应表』。那么人们就可以直接藉由主机名来连接上某些网络上的主机啰！

然而因为科技的发达，连上 Internet 的人们越来越多，使用 /etc/hosts 的方法已经搞不定了(只要一部新计算机上线，全部 Internet 上面的所有计算机都要重新改写 /etc/hosts 才行！)，这个时候领域名系统 (Domain Name System, DNS) 就适时的出现了！

DNS 利用类似树状目录的型态，将主机名的管理分配在不同层级的 DNS 服务器当中，经由分层管理，所以每一部服务器记忆的信息就不会很多，而且异动上面也相当的容易修改！ 这个 DNS 的功能你知道了吗？对啦！就是『将计算机主机的名称转译成 IP 』就是了！当然啰，他的额外功能还很多，关于 DNS 的详细的解析部分我们将在后续的 第十九章 DNS 服务器架设当中再持续的加强内容，总之， 它的最大功能就是『让有意义的，人类较容易记忆的主机名(英文字母) ，转译成为计算机所熟悉的 IP 地址！』

透过上面的简单说明，你得要知道，如果你想要一个主机名，那你就得要透过 DNS 系统，而不是单纯的修改你的 /etc/hosts 而已。 那你如何加入一个主机名到 DNS 系统当中呢？『授权』是重点！那什么是授权呢？

10.1.2 重点在合法授权

很多朋友都认为：『因为我想要架站，所以主机需要有个主机名，因此我就得要架设 DNS 服务器？』是这样吗？当然不是啰！DNS 是个很庞大的架构，而且是串连在全球的网络当中，除非你经由『注册』的手续来让 DNS 系统承认你主机名存在的合法性，否则你架设的 DNS 只能说是一个『地下练习的测试站』而已啦！并没有用途的。

那我要如何加入 DNS 系统呢？很简单啦！首先你必须要选择一个注册单位， 并且检查出你想要注册的主机名是否存在？主机名是有意义的，并不是你可以随便注册的喔！ 举例来说，在台湾常见的个人网站注册主机名为： *.idv.tw* ，而公司行号则可能注册为 .com.tw 了！这个得要特别留意。至于台湾地区的注册单位很多，你可以选择例如 Hinet 或 Seednet 之类的 ISP 来注册。当然，你也可以选择免费的 no-ip.org 来注册的。

如果想要了解啥是『合法授权』的话，得要从 DNS 主机名的查询方式来谈起，由于 DNS 查询的方式都是由上层的 ISP 提供解析授权给下游的注册者，因此，下游的注册者只要设定妥当后， 全世界的主机就会知道你设定的数据了。详细的查询流程我们留到DNS 服务器章节再来谈， 底下仅是介绍一个简单的查询示意图。

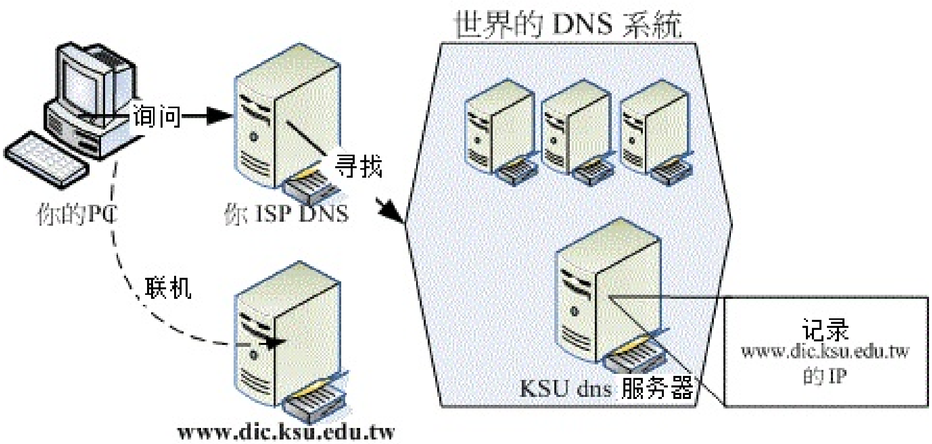


图 10.1-1、 DNS 查询示意图

举昆山信息传播系的 WWW 服务器的主机名注册方式来说好了，我们系上得要先跟昆山计中(相当于我们的 ISP) 注册取得 www.dic.ksu.edu.tw 这个主机名与 IP 的对应， 这个对应信息是写在昆山计中的 DNS 服务器上，与资传系的 WWW 服务器无关喔！那你怎么知道那部 www.dic.ksu.edu.tw 在哪里？你会先向你的 DNS 要求查询，该 DNS 会去向全世界的 DNS 系统查询，该系统会主动的查询到 KSU dns 服务器， 然后你的 PC 就会知道 www.dic.ksu.edu.tw 的 IP 在哪，最后你就开始联机啰。

从这个流程当中，你可以发现我们的 www 服务器与 KUS dns 服务器没有绝对关系，两者是独立的，我们只要作好 DNS 的『注册』工作 (向计中申请注册) 即可，并不需要去维护 DNS 的信息。所以啰， 这里你只要知道：(1)主机名的设计是有意义的，不可以随便设定、 (2)主机名要生效，得要透过注册来取得合法授权。这样就好了，如果想要架设 DNS 与更了解 DNS 系统的话， 等到后续的 DNS 服务器章节再来谈。

Tips: 在这个章节当中，理论方面的讲解比较少，因为很多数据都与 DNS 服务器篇 有重复，在这个章节当中鸟哥主要在介绍动态 IP 架站的一个简单主机名申请方式啦！ ^_^



10.1.3 申请静态还是动态 DNS 主机名

由上面的说明当中，我们可以很清楚的知道 DNS 系统最大的功能就是在主机名对应 IP 的转译上面。当然啦，预设的 DNS 转译是用在『固定 IP 对应主机名』的方法上面的！就像上面的图 10.1-1 所示一般。在这个情况底下，你在 DNS 架构下申请完主机名后， 如果你的 IP 不会更动，那就永远不用去烦恼主机名的相关问题啰，这也是所谓的静态 DNS 主机名功能。

但是...天寿喔！我们的很多小网站都是以非固定 IP 来上网的，更有甚者，某些 ADSL 拨接模式甚至会定时强制断线，也就是说， 在一段时间后，我们都得需要重新拨接上网，而每次拨接成功后取得的 IP 可不见得相同啊，如此一来 IP 不是一直在变吗？ 那么我不就需要一直跟我上层 DNS 主机的管理员申请『变更 IP 』吗？会不会太麻烦了点？

是很麻烦啊！所以现在为了解决这个问题，很多 ISP 提供了所谓的动态 DNS 服务的功能，他是这样做的：

1. Client 端(就是你啦)每次开机或者是重新拨接，并取得一个新的 IP 之后，会主动向 DNS Server 端提出要求，希望 Server 端变更主机名与 IP 的对应 (这个步骤在每个主要的 ISP 都有提供适当的程序来提供给 client 使用)；
2. Server 端接受 Client 端的要求之后，会先去查询 Client 提供的账号密码是否正确， 若正确就会立即修改 Server 本身对于你的主机名的设定值。

所以啰，每次我们取得了新的 IP 之后，我们的主机名对应的 IP 也会跟着在 DNS 系统上面更新， 如此一来，只要别人知道你的主机名，不论你的 IP 为何，他一定可以连上你的主机 (因为 IP 跟着你的主机而变！) 这对于我们这种使用动态 IP 的人是很有帮助的！ (阿！眞是造福我们这些穷苦人家的孩子呀！) 整个程序就有点像底下的图示，WWW 服务器与 DNS 服务器之间就有关连性啦。

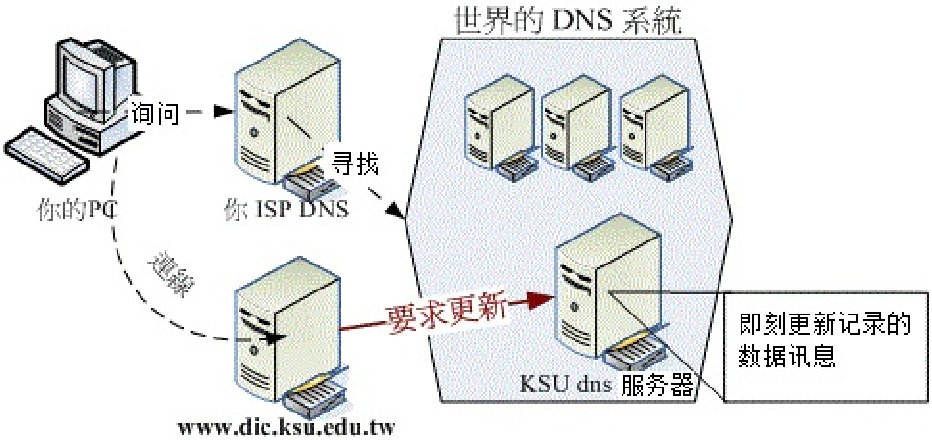


图 10.1-2、动态 DNS 服务--客户端向服务器端发送更新要求

不过，还是需要注意的是，目前的主机名申请很多是『需要钱的』！如果你需要比较稳定的主机名对应 IP 的服务，那么花点钱来注册还是必须的。不过，如果是实验性质的网站，那么也是可以申请免费的动态 DNS 服务喔！

10.2 注册一个合法的主机名

根据前面的说法，如果你只想要有合法的主机名的话，那么依据你的 IP 是否固定而有：(1) 静态 DNS 主机名与 (2)动态 DNS 主机名两种注册方式。底下鸟哥列出自己有注册经验的网站提供大家参考：

- 静态 DNS 主机名注册： 静态 IP 对应主机名的注册网站实在太多了，底下是鸟哥有注册经验的网站：
    - 台湾网络信息中心：http://www.twnic.net
    - 国外的领域名系统：http://www.netsol.com
    - 国外的领域名系统：http://www.dotster.com
    - 国外的领域名系统：http://www.godaddy.com
- 动态 DNS 主机名注册： 至于免费的动态 DNS 系统主要就是这个 NO-IP 公司提供的网站啰！如下连结：
    - 国外的免费 DNS 系统：http://www.no-ip.com

10.2.1 静态 DNS 主机名注册：以 Hinet 为例

静态 DNS 的申请方式其实都差不多，都是需要：

1. 先查询所想要注册的网域是否存在；
2. 进入 ISP 去申请注册你所想要的主机名；
3. 缴费，并等待主机名被启用。

我们以台湾蛮常见的 Hinet 这个 ISP 提供的『个人网域： .idv.tw』注册方式来说明：

- 1. 登入主画面，并查询欲注册网域是否存在

    先连结到底下的网页去：http://domain.hinet.net/ ，并在 whois 的画面当中(右上角)选择你想要注册的主机名，按下『Go』开始搜寻。

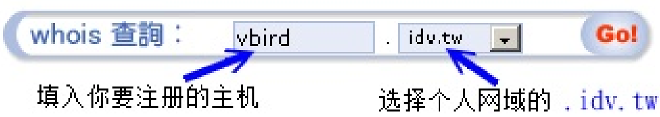
    

    图 10.2-1、利用 whois 查询欲注册网域是否存在

- 2. 逐步进行注册

如果确认你的主机名没有被注册掉，那么你就可以开始注册了！同样的在上面的网站连结当中， 选择『个人域名』就可以开始申请了！请由『域名申请』开始依序一步一步办理！这里不再说明了，反正都是中文， 看的懂得啦！ ^_^

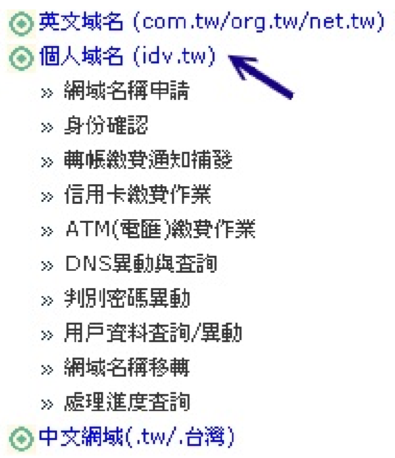


图 10.2-2、个人网域逐步注册的流程示意图

- 3. 填写主机名对应的 IP

通常花个几天等待缴费完毕后，我们就可以开始进行登入与主机名的填写了！在图 10.2-2 的图示中按下『DNS 异动与查询』的项目， 并填入当初注册时的主机名与密码，然后就会出现如下的画面了：



图 10.2-3、主机名与 IP 对应的填写范例

特别的给他留意，因为我们没有要架设 DNS 主机，所以当然最上方要选择『主机』的项目， 然后你可以填入三部主机名喔！当然，这三部主机名可以通通指向同一个 IP ，也可以不同！ 随你的便呐！需要注意到的是，你的主机名应该是『othername.yourhost.idv.tw』后面的 yourhost.idv.tw 是不变的，前面的 othername 则可以自由选取呢！例如鸟哥上面的设定，后面均是 vbird.idv.tw ，而前面的名称就可以让我自由选择啦！

- 4. 等待 DNS 启用

在上图 10.2-3 当中按下『填写完请按这里』后，就等着启用吧！不过设定成功到可以使用，其实需要一定的时间的。以鸟哥为例，第一次申请之后，大约过了 20 小时该设定才正确的启动呢！请耐心等候啊！不要太着急啰！ ^_^

在台湾，各家的领域名注册流程都差不多，不过，金额是有点差异的，当然，服务也就有不同啊！鸟哥的 vbird.org 领域名则是在 http://www.godaddy.com 注册的喔！如果你不想要使用 .idv.tw 来注册的话，那么国外的 ISP 提供的 DNS 也可以考虑看看说！

10.2.2 动态 DNS 主机名注册：以 no-ip 为例

如果你跟鸟哥一样使用 ADSL 拨接的方式来上网，这表示你的 IP 应该是不固定的！果真如此的话，那想要用这样的网络环境来架站就比较麻烦一点！因为上面利用 Hinet 注册的方式通常是给固定 IP 使用的，你应该不会想要天天上去更新你的 IP 吧？此时这个 no-ip.com 所提供的免费动态 IP 对应主机名的服务就很重要啦！我们先来申请一个主机名来玩玩吧！ ^_^

- 1. 登入主网页，并且注册一个新账号

  你必须要连上 http://www.no-ip.com 这个网站，然后在出现的画面当中的右上角部分，选择『 Create Account 』那个项目。不过，如果你已经有 no-ip 网站的注册账号，那么直接跳到底下第四步骤去登入即可。

  图 10.2-4、no-ip 网站的注册：新建账号点选

- 2. 开始填写识别数据

  由于启动账号必须由 no-ip 提供一个注册启动的连结，因此你必须要填写正确的 email 来接受启动码。整个注册的讯息如下图所示：

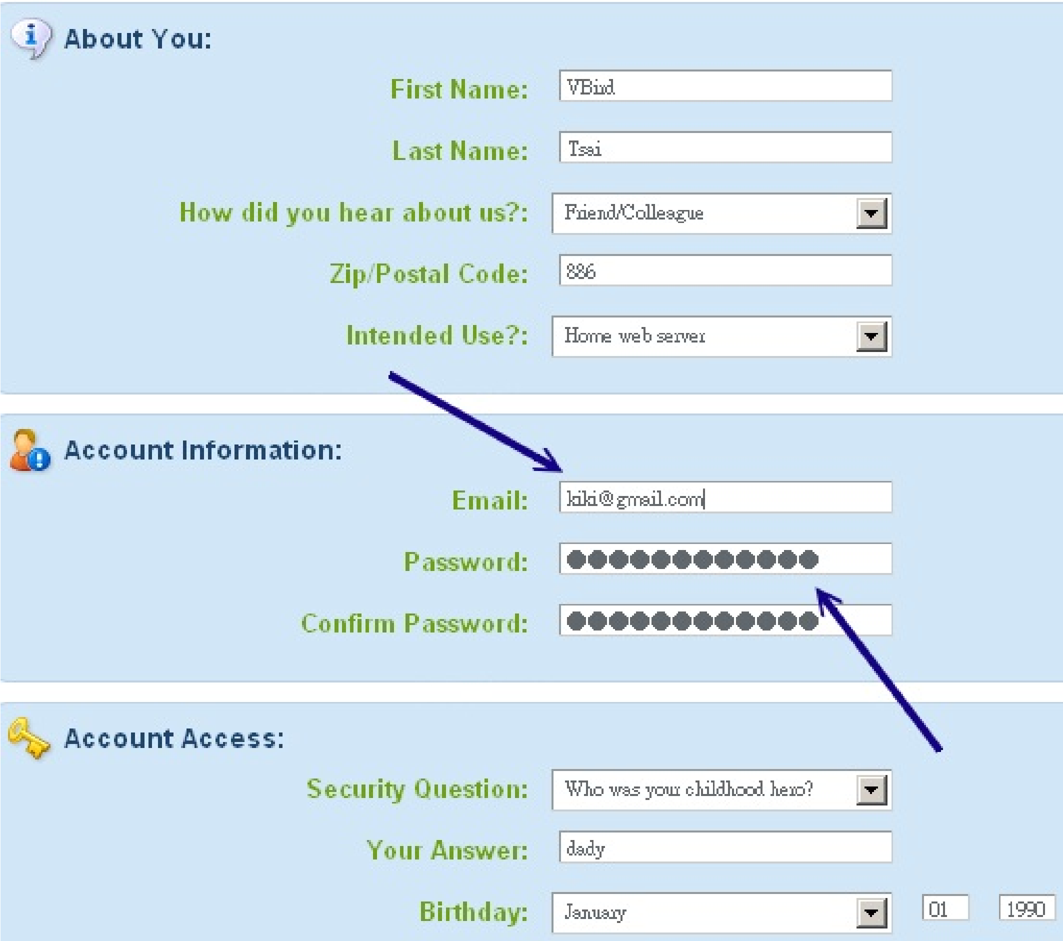


图 10.2-5、no-ip 网站的注册：新账号建立所需填写数据

最重要的是，在该网页的最下方还有验证码以及你必须要勾选的『I agree that...』项目才行喔！ 最后才点选『I Accept, Create my Account』项目噜。详细图示如下所示：

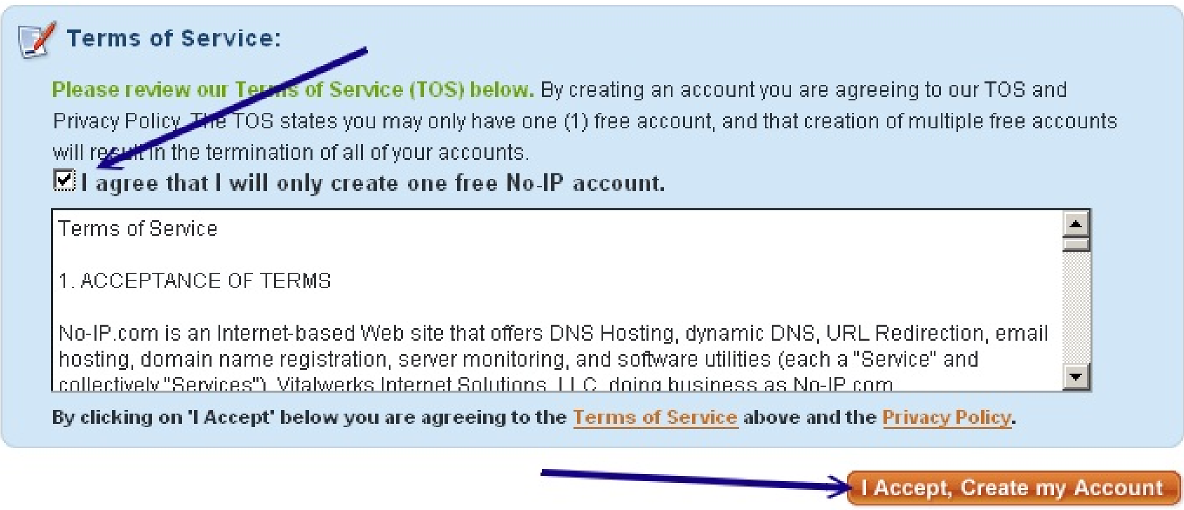


图 10.2-6、no-ip 网站的注册：新账号建立务必勾选项目

- 3. 启用账号

在你申请注册一个新账号后， no-ip 会发一封信给你，请自行参考信件内容，并点选正确的启动码连结， 那你的账号就能够启动，此时请回到图 10.2-4 去，针对你的 email(username)/密码(password) 填写妥当，就能够登入 NO IP 网站了。

- 4. 登入 no-ip 且设定主机名与 IP 的对应

透过图 10.2-4 的样子来登入后，你会看到有点像底下的图示，底下就准备来处理你主机名与 IP 的对应数据了：

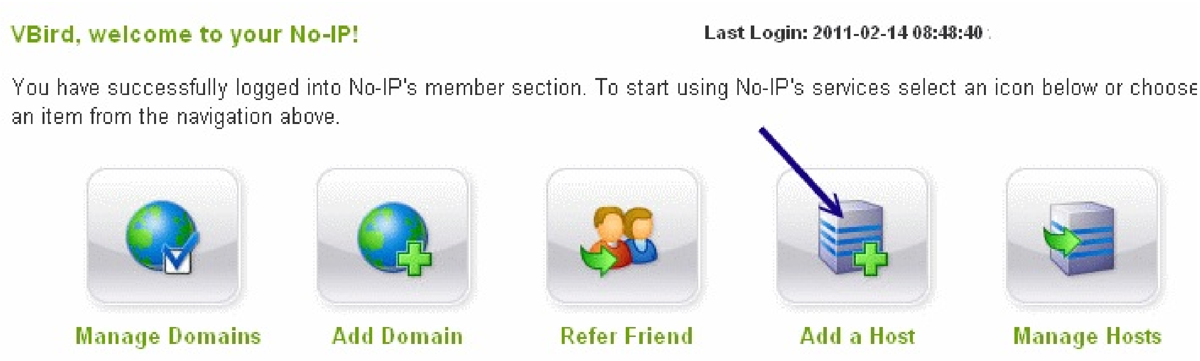


图 10.2-7、登入 NO IP 网站后的示意图

上图的重点在于『Add a Host (新增一个主机名)』及『Manage Hosts (管理主机名)』两者，由于我们都还没有主机名的设定， 因此首先就使用 Add a Host 来新增一笔主机名吧！按下那个图示，之后就会出现底下的画面：

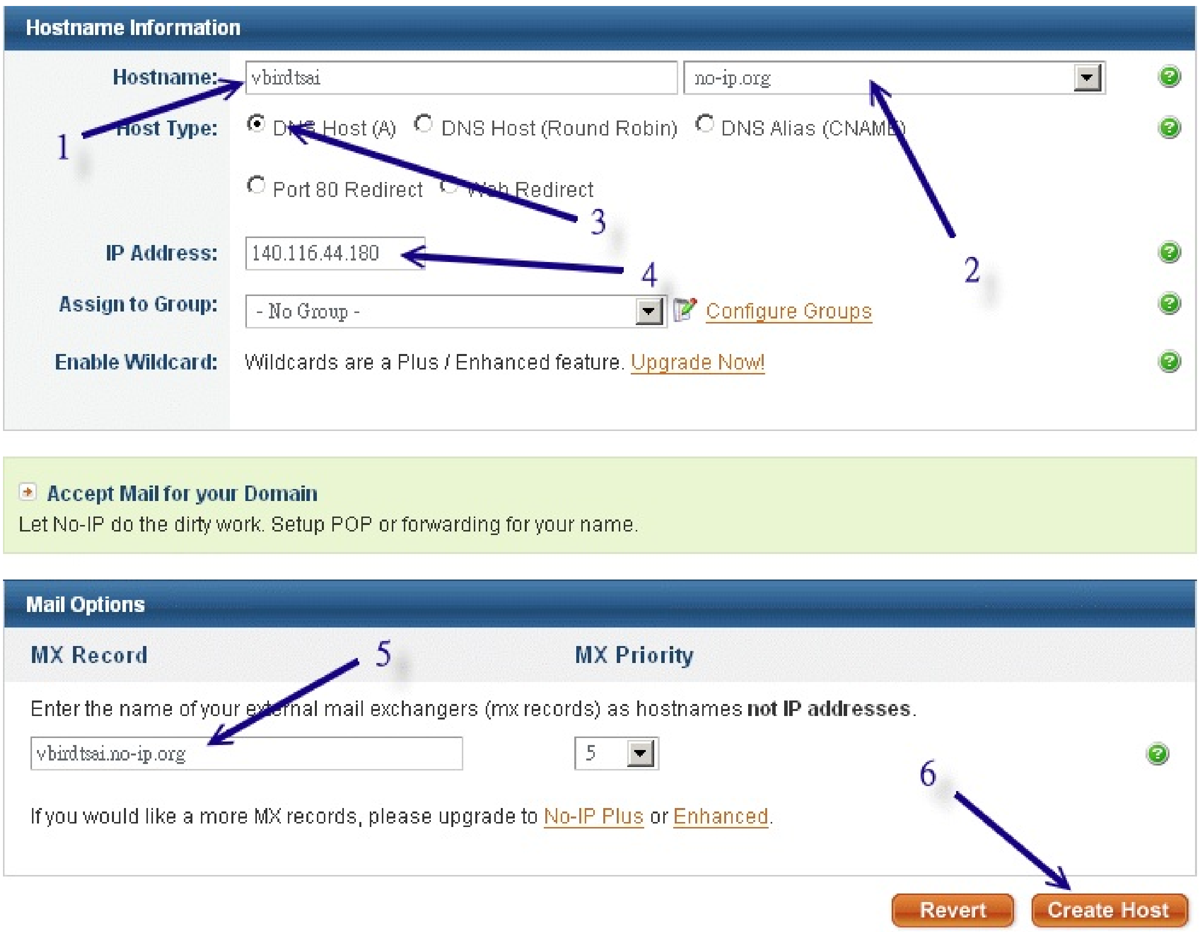


图 10.2-8、新增一个主机名与 IP 对应的方式

主要填写的内容为：

1. 你想要的主机名；
2. No IP 网站提供的领域名，与上个名称组合成完整的主机名；
3. 选择单一主机的 IP 对应；
4. 填写该主机名对应的正确 IP 为何 (后续可以透过程序直接修改，这里随便填也没关系)
5. 只与 mail server 有关，所以写不写都无所谓，不过，建议填写自己的主机名即可
6. 若上述数据都正确，按下 Create Host 即可建立成功。如果该主机名有被使用掉的话，屏幕会出现警告讯息， 此时请再选填另外的主机名吧！ 如果一切都没有问题的话，应该就会出现如下所示的图示。未来如果你想要更新或者是删除或者是新增主机名的话， 就透过下图的示意流程来处理即可。且由下图你也可以知道， NO IP 有提供 5 个免费的主机名给你使用喔！眞是太棒了！ 如果你想要维护相关数据，就使用『 Manage Hosts 』按钮即可处理了。



图 10.2-9、主机名处理完毕与维护的示意画面

- 5. 设定自动更新主机名与 IP 的对应

如果系统重新启动，或者是重新拨接取得一个新的 IP 后，我们都要登入 no-ip 网站来修改的话， 那就太没有效率了！所以 no-ip 提供一个好用的客户端程序给系统管理员使用，你可以在 no-ip 官网右上方的『Download』 处选择相关的档案。该网站目前提供给 Linux, Windows 与 MAC 等系统使用的程序，非常方便。 我们当然是选择 Linux 那个项目啊！请自行下载并且将该程序移动到 Linux 系统上吧！ 整个安装与启用的流程式这样的：

```
# 1\. 编译与安装：
[root@www ~]# wget  \
&gt; http://www.no-ip.com/client/linux/noip-duc-linux.tar.gz
[root@www ~]# cd /usr/local/src
[root@www src]# tar -zxvf /root/noip-duc-linux.tar.gz

[root@www src]# cd noip-*
# 注意一下，这个目录里面有个文件名为 README.FIRST 的档案，务必察看一下内容！
[root@www noip]# make
[root@www noip]# make install
# 这样会将主程序安装在 /usr/local/bin/noip2 而主参数档放在
# /usr/local/etc/no-ip2.conf 当中！然后你必须要开始回答一些问题：

Please select the Internet interface from this list.

By typing the number associated with it.
0       eth0
1       eth1
0      &lt;==因为鸟哥的主机对外使用 eth0 接口

Please enter the login/email string for no-ip.com  kiki@gmail.com
Please enter the password for user 'kiki@gmail.com' ***
# 上面这两个是你刚刚注册时所填写的 email 与密码喔！

Only one host [vbirdtsai.no-ip.org] is registered to this account.
It will be used.
Please enter an update interval:[30]
Do you wish to run something at successful update?[N] (y/N) n

mv /tmp/no-ip2.conf /usr/local/etc/no-ip2.conf
# 重点在此！刚刚你做的配置文件被放到上面这个档案中了！
```

这样就将你的 no-ip 制作完毕，而且也可以开始来执行啰！执行的方法也是很简单啦！

```
# 2\. noip2 的程序使用：
[root@www ~]# /usr/local/bin/noip2
# 不要怀疑！这样输入后，你在 no-ip 上面注册的主机名，
# 就开始可以自动的产生对应了！就这么简单！

[root@www ~]# noip2 [-CS]
选项与参数：
-C ：重新设定参数，亦即设定刚刚我们上面输入粗体字的咚咚！
     如果你有两个以上的 no-ip 主机名时，就一定需要使用 noip2 -C
     来重新设定参数档案！
-S ：将目前的 noip2 的状况显示出来！

[root@www ~]# noip2 -S
1 noip2 process active.

Process 2496, started as /usr/local/src/noip-2.1.9-1/noip2, (version 2.1.9)
Using configuration from /usr/local/etc/no-ip2.conf
Last IP Address set 140.116.44.180
Account kiki@gmail.com
configured for:
        host  vbirdtsai.no-ip.org
Updating every 30 minutes via /dev/eth0 with NAT enabled.
```

嘿嘿！这样就成功了！而且每 30 分钟 noip2 可以自动的去主网站上面进行更新呢！真是很不错！那如果想要一开机就启动 noip2 呢？这样做即可：

```
# 3\. 设定开机启动：
[root@www ~]# vim /etc/rc.d/rc.local
# 加入底下这一行：
/usr/local/bin/noip2
```

10.3 重点回顾

- 主机名的目的在辅助人们记忆 TCP/IP 的 IP 数值；
- 主机名与 IP 的对应，由早期的 /etc/hosts 变更为 DNS 系统来记录
- 合法的主机名必须要透过合法授权后，才能够在 Internet 上面完整的生效
- 除了静态的主机名与 IP 对应外，若是不固定 IP 的联机模式，可以透过动态 DNS 服务，来达成非固定 IP 永远指向同一个主机名的任务。

10.4 本章习题

- 请简易说明 /etc/hosts 的用途；这个档案是早期用在进行主机名与 IP 的解析的，目前比较常用在内部私有网域的名称解析上，可以加快内部网域的反查喔！
- 请说明『合法授权』的主机名需要做什么？如果想要合法授权，就需要向上层 DNS 主机『注册』才行！而且还要上层 DNS 主机管理员愿意将领域名的解析权限授权给你啊！
- 什么是动态 DNS 系统？(仅说明 client 端)因为我们的 Client 拨接时，得到的 IP 都不是固定的，所以无法以 DNS 系统进行固定 IP 对应主机名的工作！ 此时就需要动态 DNS 系统了！以 DNS 主机提供的动态更新主机名对应 IP 的机制，可以让我们的不同 IP 对应到同一个主机名呐！
- 如果你使用 adsl 拨接来上网设定服务器，那么该申请哪一类型的主机名？为什么？因为我是以 ADSL 上网拨接，所以 IP 是不固定的，此时需要申请动态 DNS 主机的主机名，例如 no-ip.org 等等！

10.5 参考数据与延伸阅读

- 台湾网络信息中心：http://www.twnic.net/
- 国外的领域名系统：http://www.netsol.com/
- 国外的领域名系统：http://www.dotster.com/
- 国外的免费 DNS 系统：http://www.no-ip.com

2002/08/05：第一次完成日期！ 2003/08/26：修改篇名、增加一 no-ip 的设定方式！ 2006/09/15：将旧的文章移动到 此处。 2011/02/09：将基于 CentOS 4.x 所写的文章移动到 此处 2011/02/15：由于 noip 网站页面有变更，作了一些图片的重新处理，其他则是大同小异。 2011/07/23：将基于 CentOS 5.x 的版本移动到此处

第三部分：局域网络内常见的服务器架设

在开始实际 Linux 的网络服务器架设之前，切记前面两篇的内容都已经阅读过，并且已经强化您主机的网络安全后，才开始架设吧！ 老实说，台湾中小企业非常多，企业人员常常需要共享一些数据，这些数据当然不会传送到因特网上，而是在内部局域网络上传输的。 因此，局域网络内部的服务器架设量可能比我们因特网上面的服务器还要多呢！所以说，我们先来了解一下局域网络内常见的服务器吧。

在这一篇当中，我们主要会介绍管理服务器用的联机服务器如 ssh, xdmcp, VNC, xrdp 等服务，然后针对网络参数管理部份来介绍 DHCP 服务器，在针对文件服务器的 NFS, SAMBA 等服务，再介绍与 NFS 相关性很强的账号同步的 NIS 服务器，然后介绍两个常见的服务器，包括时间同步的 NTP 与网络监控能力较佳的 Proxy 服务器，最终再讲一个磁盘不够时的 iSCSI 仿真器吧！

第十一章、远程联机服务器SSH / XDMCP / VNC / RDP

最近更新日期：2011/11/24

维护网络服务器最简单的方式不是跑去实体服务器前面登入，而是透过远程联机服务器联机功能来登入主机， 然后再来进行其他有的没的维护就是了。Linux 主机几乎都会提供 sshd 这个联机服务，而且这个服务还是主动进行数据加密的！ 讯息在网络上面跑安全多了。同时我们还能透过 rsync 这个指令以 sshd 信道来达成异地数据备援的功能哩！相当不赖。 如果想要利用图形接口登入，那么默认的 Xdmcp 配合 VNC 就能够使用图形接口在网络的另一端登入你的服务器！ 如果你习惯使用 Windows 的远程桌面，那么 XRDP 也不要放过啰！

11.1 远程联机服务器

远程联机服务器对我们来说，可是一项很有用的工具啊！他可以让我们更方便的管理主机。不过，方便归方便，但开放全世界都可以尝试登入你的主机并不个好主意，因为可能会有安全性的问题呐！ 所以本章才要特别强调一下这个玩意儿啊！

11.1.1 什么是远程联机服务器

首先，我们来了解一下，什么是『远程联机服务器』？ 这个东西的功能为何？我想，你应该已经听过，一部开放到因特网上的服务器，基本上，它可以不需要屏幕、键盘、 鼠标等等的周边配备，只要有基本的主板、CPU、RAM、硬盘再加上一块好一点的网络卡，并且连上因特网， 那这部主机就能够提供你有需要的网络服务了。但如果你需要重新设定这部主机，该如何登入主机取得类似 bash 的接口来操纵与进行修改呢？那就得要透过联机服务器的服务了。

是的！你猜对啦，远程联机服务器在提供你由远程透过文字或图形接口的方式来登入系统，让你在远程的工作机前面登入 Linux 主机以取得可操控主机之接口 (shell)，而登入后的操作感觉上就像坐在系统前面一样！ 所以啦，你当然不需要远程网络服务器的键盘、鼠标、屏幕等等。你只要工作机可以正常联机到远程主机即可啊。

以鸟哥个人为例，目前鸟哥管理十几部的 Unix-Like 主机，这些主机都不放在同一个地方，分布在南台湾各处！ 那么当有新的软件的漏洞被发布，或者是需要进行一些额外的设定的时候，是否鸟哥本人一定要到现场吗？当然不需要， 只要透过网络联机到该主机上面，就可以进行任何工作了！真的就好像在主机前面工作一般的轻松愉快！ ^_^！这就是远程联机服务器啦！

Tips: 很多人会说，我用 FTP 也要输入账号密码来登入啊？那与这个章节谈到的登入有何不同？最大的不同在于取得的 shell 能进行的工作啦！用 ssh/telnet/VNC 等方式取得的文字或图形 shell 能够进行很多系统管理的任务，与单纯的 FTP 能进行的工作当然不同！



- 联机服务器的功能作用之一：分享 Unix Like 主机的运算能力

当你的工作需要使用到 Linux 强大的程序语言编译功能时，那么你一定需要 Linux 对吧！而且最好是指令周期快一点的主机， 这个时候你可以将你研究室最快的那一部主机开放出来，设定一下远程联机服务器，让你的学生啦，或者是研究室的同仁啦， 可以透过这部机器帮他们

进行研究的工作，这个时候，你的主机就可以让多人进行分享 Linux 运算的功能啦！

举例来说，鸟哥与昆山还有长荣大学的老师、同学们组建了一组服务器等级的丛集架构计算机 (PC cluster)， 目前我们在该计算机上面跑 MM5 、Models3 等大气与空气质量模式，要在这样的架构底下跑数值模式的原因， 主要就是考虑运算能力。那会使用到该组计算机的有好多人，难道大家都在挤在一部屏幕前面工作？当然不需要啦！ 这时候就是远程联机服务器的服务范围啰！

但是否每一部连到 Internet 上面的主机都应该要开放远程联机的功能呢？其实并不尽然， 还是需要针对你的主机来进行规划的，我们底下分服务器与工作站来说明：

- 服务器类型 (Server) ：有限度的开放联机

在一般对因特网开放服务的服务器中，由于开放的服务可能会有较为重要的信息，而远程联机程序连进主机之后， 可以进行的工作又太多了(几乎就像在主机前面工作一般！)，因此服务器的远程联机程序通常仅针对少部分系统维护者开放而已！ 除非必要，否则 Server 类型的主机还真的不建议开放联机的服务呢！

以鸟哥为例，我的主机提供了我们研究室使用 Mail 与 Internet 上面的 WWW 服务，如果还主动提供远程联机的话， 那么万一不小心被入侵，那可就伤脑筋了！因此，鸟哥仅开放『很小部分的网域』让系统管理员连进来， 其他来源的 IP 一律抵挡！不许使用远程联机的功能呢！

- 工作站类型 (Workstation) ：只对内网开放

所谓的工作站就是不提供因特网服务的主机，仅提供大量的运算能力给使用者。 既然不提供因特网的服务，那你还开联机服务器干嘛？不是啦！像前面鸟哥提到的 PC cluster 大量运算的整组计算机， 也可以称之为工作站，因为它没有提供常见的网络服务嘛！不过必须要提供给使用者登入的权限，这样大家才用的到运算功能啊！ 此时你就得要针对内部，或者是特定的某些来源开放他们使用你的工作站啰！

11.1.2 有哪些可供登入的类型？

那么目前远程联机服务器的主要类型有哪些？如果以登入的联机界面来分类，基本上有文字接口与图形接口两种：

- 文字接口明码： telnet, rsh 等为主，目前非常少用；
- 文字接口密码： ssh 为主，已经取代上述的 telnet, rsh 等明码方式；
- 图形接口： Xdmcp, VNC, RDP 等较为常见

在文字接口登入的联机服务器，主要有以『明码』传送数据的 telnet 服务器，及以加密技术进行数据加密再传送的 SSH 服务器！虽然 telnet 可以支持的客户端软件比较多，不过由于它是使用明码来传送数据，你的数据很容易遭到有心人士的撷取！ 所以近来我们都呼吁大家多使用 SSH 这一种联机方式

至于图形接口的联机服务器，比较简单的有 Xdmcp (X Display Manager Control Protocol)，架设 Xdmcp 很简单， 不过客户端的软件比较少。另外一款目前很常见的图形联机服务器，就是 VNC (Virtual Network Computing)， 透过 VNC server/client 软件来进行连接。如果你想要使用类似 Windows 的远程桌面联机，该功能使用的是 RDP (Remote Desktop Protocol)，那你可得要架设 RDP 服务器才行。

Tips: 图形接口最大的优点是『图形』啊！不过，因为是透过图形来传送，传输的数据量相当的大， 所以速度与安全性都有待考虑。因此，我们仅建议你将图形接口的远程登录服务器开放在内部网域 (LAN) 就好了！



- 数据传送的明码与密码

什么是『明码』与『加密』的数据封包传送模式呢？为什么 telnet 使用明码就比较不安全？所谓的明码就是：『当我们的数据封包在网络上传输时，该数据封包的内容为数据的原始格式』， 也就是说，你使用 telnet 登入远程主机时，不是得要输入账号密码吗？那你的账号密码是以原本的数据格式传输， 所以如果被类似 tcpdump 之类的监听软件撷取数据， 那你的帐密就有可能被窃取啦！

所以啦，万一你的数据封包里面含有信用卡数据、密码、身份确认等重要信息时，是否很危险呐？ 因此，目前我们通常都希望使用可以将这些在网络上面跑的数据加密的技术，以增加数据在 Internet 上面传送的安全性啊！

Tips: 说 ssh 比较安全，其实是透过 ssh 信道传输讯息时，该讯息在网络上面比较安全，因为数据是加密过的，即使被窃取， 对方可能也不会知道数据内容为何，因此信息比较安全。但这不代表 ssh 这个通讯协议就比较安全喔！两者意义不同！



由于明码传输的 telnet, rsh 等联机服务器已经被 ssh 取代，并且在一些实际应用上已经很少看到 telnet 与 rsh 了， 因此本章在文字接口上着重于介绍 ssh 的应用，包括以 rsync 藉由 ssh 通道来进行异地备援的任务等等。至于图形接口则会介绍 Xdmcp, VNC 与 RDP 喔！因为很多工作站用户需要显示他们在工作站实作后的图形呈现，因此这部分也是很重要的呢！

11.2 文字接口联机服务器： SSH 服务器

由于先前的远程联机服务器大多是明码，而且协议也有些资安问题，因此后来就有 SSH 这个协议来取代上述这些咚咚。 那么 SSH 是什么呢？它有什么特异功能？简单的来说，SSH 是 **S**ecure **SH**ell protocol 的简写 (安全的壳程序协议)，它可以透过数据封包加密技术，将等待传输的封包加密后再传输到网络上， 因此，数据讯息当然就比较安全啰！这个 SSH 可以用来取代较不安全的 finger, R Shell (rcp, rlogin, rsh 等), talk 及 telnet 等联机模式。底下我们将先简介一下 SSH 的联机模式，来说明为什么 SSH 的数据讯息会比较安全呢！

特别注意：这个 SSH 协议，在预设的状态中，本身就提供两个服务器功能：

1. 一个就是类似 telnet 的远程联机使用 shell 的服务器，亦即是俗称的 ssh ；
2. 另一个就是类似 FTP 服务的 sftp-server ！提供更安全的 FTP 服务。

11.2.1 联机加密技术简介

什么是『数据加密』呢？简单的说，就是将人们看的懂得原始电子数据，经过一些运算，让这些数据变成没有意义的乱码 (至少对人类来说)，然后再让这个咚咚在网络上面传输，而当用户想要查阅这个数据时，再透过解密运算， 将这些咚咚反推出原始的电子数据。由于这些数据已经被重新处理过，所以，即使数据在因特网上被 cracker 监听而窃取，他们也不容易就推算得出来原始资料内容的。

Tips: 鸟哥常常说，加密机制有点像是两个人之间的火星语对话啦！如果你跟你的朋友约定好使用你们制订的某种特别语言， 这个语言只对你们两个有意义。那么当你们两人讲话时，在旁边的人听到的只是一堆没有意义的声音，因为他们听不懂啊！ 即使路人将你的声音录下来，只要他不知道你们的特殊用语，那他就不可能了解你们对话的内容啰。



加解密运算的机制与技术非常多，我们这里不去讨论复杂的理论问题，只谈对我们比较有关的一些加解密概念而已。 目前常见的网络封包加密技术通常是藉由所谓的『非对称密钥系统』来处理的。 主要是透过两把不一样的公钥与私钥 (Public and Private Key) 来进行加密与解密的过程。由于这两把钥匙是提供加解密的功用， 所以在同一个方向的联机中，这两把钥匙当然是需要成对的！它的功用分别如下：

- 公钥 (public key)：提供给远程主机进行数据加密的行为，也就是说，大家都能取得你的公钥来将数据加密的意思；

- 私钥 (private key)：远程主机使用你的公钥加密的数据，在本地端就能够使用私钥来进行解密。由于私钥是这么的重要， 因此私钥是不能够外流的！只能保护在自己的主机上。

由于每部主机都应该有自己的密钥 (公钥与私钥)，且公钥用来加密而私钥用来解密， 其中私钥不可外流。但因为网络联机是双向的，所以，每个人应该都要有对方的『公钥』才对！那如果以 ssh 这个通讯协议来说，在客户端与服务器端的相对联机方向上，应该有如下的加密动作：

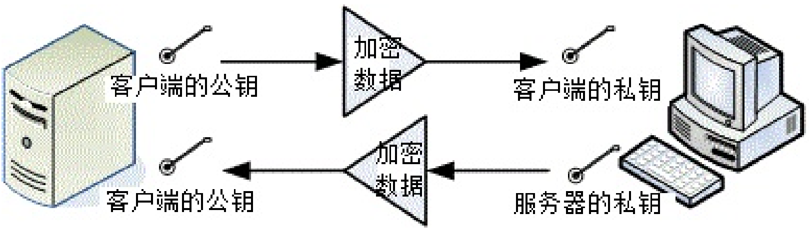


图 11.2-1、公钥与私钥在进行数据传输时的角色示意图

如上图所示，我们如果站在客户端的角度来看，那么，首先你必须要取得服务器端的公钥，然后将自己的公钥发送给服务器端， 最终在客户端上面的密钥会是『服务器的公钥加上客户端我自己的私钥』来组成的。

Tips: 数据加密的技术真的相当的多，也各有其优缺点，有的指令周期快，但是不够安全；有的够安全，但是加密/解密的速度较慢～ 目前在 SSH 使用上，主要是利用 RSA/DSA/Diffie-Hellman 等机制喔！



目前 SSH 的协议版本有两种，分别是 version 1 与 version 2 ，其中 V2 由于加上了联机检测的机制， 可以避免联机期间被插入恶意的攻击码，因此比 V1 还要更加的安全。所以啰，请尽量使用 V2 版本即可，不要使用 V1 啰。 无论是哪种版本，都还是需要公私钥加密系统的，那么这些公钥与私钥是如何产生的呢？底下我们就来谈一谈啦！

- SSH 的联机行为简介

我们可以将 ssh 服务器端与客户端的联机步骤示意为下图，至于步骤说明如后：

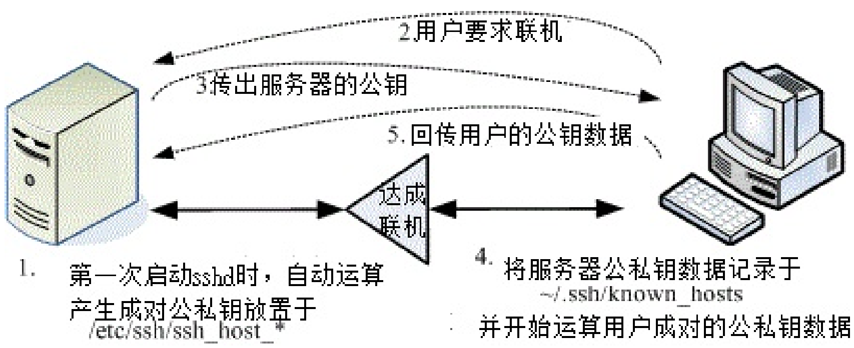


图 11.2-2、ssh 服务器端与客户端的联机步骤示意图

1. 服务器建立公钥档： 每一次启动 sshd 服务时，该服务会主动去找 /etc/ssh/ssh_host* 的档案，若系统刚刚安装完成时，由于没有这些公钥档案，因此 sshd 会主动去计算出这些需要的公钥档案，同时也会计算出服务器自己需要的私钥档；
2. 客户端主动联机要求： 若客户端想要联机到 ssh 服务器，则需要使用适当的客户端程序来联机，包括 ssh, pietty 等客户端程序；
3. 服务器传送公钥档给客户端： 接收到客户端的要求后，服务器便将第一个步骤取得的公钥档案传送给客户端使用 (此时应是明码传送，反正公钥本来就是给大家使用的！)；
4. 客户端记录/比对服务器的公钥数据及随机计算自己的公私钥： 若客户端第一次连接到此服务器，则会将服务器的公钥数据记录到客户端的用户家目录内的 ~/.ssh/known_hosts 。若是已经记录过该服务器的公钥数据，则客户端会去比对此次接收到的与之前的记录是否有差异。若接受此公钥数据， 则开始计算客户端自己的公私钥数据；
5. 回传客户端的公钥数据到服务器端： 用户将自己的公钥传送给服务器。此时服务器：『具有服务器的私钥与客户端的公钥』，而客户端则是：『具有服务器的公钥以及客户端自己的私钥』，你会看到，在此次联机的服务器与客户端的密钥系统 (公钥+私钥) 并不一样，所以才称为非对称式密钥系统喔。
6. 开始双向加解密： (1)服务器到客户端：服务器传送数据时，拿用户的公钥加密后送出。客户端接收后，用自己的私钥解密； (2)客户端到服务器：客户端传送数据时，拿服务器的公钥加密后送出。服务器接收后，用服务器的私钥解密。

在上述的第 4 步骤中，客户端的密钥是随机运算产生于本次联机当中的，所以你这次的联机与下次的联机的密钥可能就会不一样啦！ 此外在客户端的用户家目录下的 ~/.ssh/known_hosts 会记录曾经联机过的主机的 public key ，用以确认我们是连接上正确的那部服务器。

例题：如何产生新的服务器端的 ssh 公钥与服务器自己使用的成对私钥？ (注：注意，本例题不要在已经正常运作的网络服务器上面，因为可能会造成其他客户端的困扰！)答：由于服务器提供的公钥与自己的私钥都放置于 /etc/ssh/ssh_host* ，因此你可以这样做：

```
[root@www ~]# rm /etc/ssh/ssh_host*  &lt;==删除密钥档
[root@www ~]# /etc/init.d/sshd restart
正在停止 sshd:                                [  确定  ]
正在产生 SSH1 RSA 主机密钥:                   [  确定  ] &lt;==底下三个步骤重新产生密钥！
正在产生 SSH2 RSA 主机密钥:                   [  确定  ]
正在产生 SSH2 DSA 主机密钥:                   [  确定  ]
正在激活 sshd:                                [  确定  ]
[root@www ~]# date; ll /etc/ssh/ssh_host*
Mon Jul 25 11:36:12 CST 2011
-rw-------. 1 root root  668 Jul 25 11:35 /etc/ssh/ssh_host_dsa_key
-rw-r--r--. 1 root root  590 Jul 25 11:35 /etc/ssh/ssh_host_dsa_key.pub
-rw-------. 1 root root  963 Jul 25 11:35 /etc/ssh/ssh_host_key
-rw-r--r--. 1 root root  627 Jul 25 11:35 /etc/ssh/ssh_host_key.pub
-rw-------. 1 root root 1675 Jul 25 11:35 /etc/ssh/ssh_host_rsa_key
-rw-r--r--. 1 root root  382 Jul 25 11:35 /etc/ssh/ssh_host_rsa_key.pub
# 看一下上面输出的日期与档案的建立时间，刚刚建立的新公钥、私钥系统！
```

11.2.2 启动 SSH 服务

事实上，在我们使用的 Linux 系统当中，默认就已经含有 SSH 的所有需要的软件了！这包含了可以产生密码等协议的 OpenSSL 软件与 OpenSSH 软件 (注1)，所以呢，要启动 SSH 真的是太简单了！就直接给他启动就是了！此外，在目前的 Linux Distributions 当中，都是预设启动 SSH 的，所以一点都不麻烦，因为不用去设定，他就已经启动了！ 哇！真是爽快～无论如何，我们还是得说一说这个启动的方式吧！直接启动就是以 SSH daemon ，简称为 sshd 来启动的，所以，手动可以这样启动：

```
[root@www ~]# /etc/init.d/sshd restart
[root@www ~]# netstat -tlnp &#124; grep ssh
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address   Foreign Address   State    PID/Program name
tcp        0      0 :::22           :::*              LISTEN   1539/sshd
```

需要注意的是，SSH 不但提供了 shell 给我们使用，亦即是 ssh protocol 的主要目的，同时亦提供了一个较为安全的 FTP server ，亦即是 ssh-ftp server 给我们当成是 FTP 来使用！所以，这个 sshd 可以同时提供 shell 与 ftp 喔！而且都是架构在 port 22 上面的呢！所以，底下我们就来提一提，那么怎么样由 Client 端连接上 Server 端呢？同时，如何以 FTP 的服务来连接上 Server 并且使用 FTP 的功能呢？

11.2.3 ssh 客户端联机程序 - Linux 用户

如果你的客户端是 Linux 的话，那么恭喜你了，预设的情况下，你的系统已经有底下的所有指令，可以不必安装额外的软件喔！ 底下就来介绍一下这些指令吧！

- ssh ：直接登入远程主机的指令

SSH 在 client 端使用的是 ssh 这个指令，这个指令可以指定联机的版本 (version1, version2)， 还可以指定非正规的 ssh port (正规 ssh port 为 22)。不过，一般的用法可以使用底下的方式：

```
[root@www ~]# ssh [-f] [-o 参数项目] [-p 非正规埠口] [账号@]IP [指令]
选项与参数：
-f ：需要配合后面的 [指令] ，不登入远程主机直接发送一个指令过去而已；
-o 参数项目：主要的参数项目有：
    ConnectTimeout=秒数 ：联机等待的秒数，减少等待的时间
    StrictHostKeyChecking=[yes&#124;no&#124;ask]：预设是 ask，若要让 public key
           主动加入 known_hosts ，则可以设定为 no 即可。
-p ：如果你的 sshd 服务启动在非正规的埠口 (22)，需使用此项目；
[指令] ：不登入远程主机，直接发送指令过去。但与 -f 意义不太相同。

# 1\. 直接联机登入到对方主机的方法 (以登入本机为例)：
[root@www ~]# ssh 127.0.0.1
The authenticity of host '127.0.0.1 (127.0.0.1)' can't be established.
RSA key fingerprint is eb:12:07:84:b9:3b:3f:e4:ad:ba:f1:85:41:fc:18:3b.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '127.0.0.1' (RSA) to the list of known hosts.
root@127.0.0.1's password: &lt;==在这里输入 root 的密码即可！
Last login: Mon Jul 25 11:36:06 2011 from 192.168.1.101
[root@www ~]# exit  &lt;==离开这次的 ssh 联机
# 由于 ssh 后面没有加上账号，因此预设使用当前的账号来登入远程服务器
```

一般使用 ssh 登入远程主机，都会填写『 ssh 账号@主机IP 』的格式， 意思是说，使用该主机的某账号登入的意思。但是很多朋友都不喜欢写账号，亦即使用『 ssh 主机IP 』的格式。如同上面的范例情况。要注意喔，如果不写账号的话，那么会以本地端计算机的账号来尝试登入远程。 也就是说，如果近端与远程具有相同的账号，那么不写账号也没有关系，如上表中的范例。但是，为了以后习惯着想， 还是一开始就使用类似 email 的方式来登入远程主机，这样的行为习惯比较好啦！

上面出现的讯息中，开头 RSA 的那行后面接的就是远程服务器的公钥指纹码，如果确定该指纹码没有问题，那么你就得要输入 yes 来将该指纹码写入服务器公钥记录文件 (~/.ssh/known_hosts)，以方便未来比对该服务器的正确性之用。 注意是要写 yes 喔，单纯输入 Y 或 y 是不会被接受的～此外， 由于该主机的公钥已经被记录，因此未来重复使用 ssh 登入此主机时，就不会出现这个指纹码提示了。

```
# 2\. 使用 student 账号登入本机
[root@www ~]# ssh student@127.0.0.1
student@127.0.0.1's password:
[student@www ~]$ exit
# 由于加入账号，因此切换身份成为 student 了！另外，因为 127.0.0.1 曾登入过，
# 所以就不会再出现提示你要增加主机公钥的讯息啰！

# 3\. 登入对方主机执行过指令后立刻离开的方式：
[root@www ~]# ssh student@127.0.0.1 find / &&gt; ~/find1.log
student@localhost's password:
# 此时你会发现怎么画面卡住了？这是因为上头的指令会造成，你已经登入远程主机，
# 但是执行的指令尚未跑完，因此你会在等待当中。那如何指定系统自己跑？

# 4\. 与上题相同，但是让对方主机自己跑该指令，你立刻回到近端主机继续工作：
[root@www ~]# ssh -f student@127.0.0.1 find / &&gt; ~/find1.log
# 此时你会立刻注销 127.0.0.1 ，但 find 指令会自己在远程服务器跑喔！
```

上述的范例当中，第 4 个范例最有用！如果你想要让远程主机进行关机的指令，如果不加上 -f 的参数， 那你会等待对方主机关机完毕再将你踢出联机，这比较不合理。因此，加上 -f 就很重要～因为你会指定远程主机自己跑关机， 而不需要在空空等待。例如：『ssh -f root@some_IP shutdown -h now 』之类的指令啰。

```
# 5\. 删除掉 known_hosts 后，重新使用 root 联机到本机，且自动加上公钥记录
[root@www ~]# rm ~/.ssh/known_hosts
[root@www ~]# ssh -o StrictHostKeyChecking=no root@localhost
Warning: Permanently added 'localhost' (RSA) to the list of known hosts.
root@localhost's password:
# 如上所示，不会问你 yes 或 no 啦！直接会写入 ~/.ssh/known_hosts 当中！
```

鸟哥上课常常使用 ssh 联机到同学的计算机去看他有没有出错，有时候会写 script 来进行答案侦测。 此时如果每台计算机都在主动加上公钥文件记录，都得要输入『 yes 』，会累死！那么加上这个 StrictHostKeyChecking=no 就很有帮助啦！他会不询问自动加入主机的公钥到档案中，对于一般使用者帮助不大，对于程序脚本来说， 这玩意儿可就很不错用了！

- 服务器公钥记录文件： ~/.ssh/known_hosts

当你登入远程服务器时，本机会主动的用接收到的服务器的 public key 去比对 ~/.ssh/known_hosts 有无相关的公钥， 然后进行底下的动作：

- 若接收的公钥尚未记录，则询问用户是否记录。若要记录 (范例中回答 yes 的那个步骤) 则写入 ~/.ssh/known_hosts 且继续登入的后续工作；若不记录 (回答 no) 则不写入该档案，并且离开登入工作；
- 若接收到的公钥已有记录，则比对记录是否相同，若相同则继续登入动作；若不相同，则出现警告信息， 且离开登入的动作。这是客户端的自我保护功能，避免你的服务器是被别人伪装的。

虽然说服务器的 ssh 通常可能会改变，问题是，如果是测试用的主机，因此常常在重新安装，那么服务器的公钥肯定经常不同， 果真如此的话，你就无法继续登入了！那怎办？让我们来模拟一下这个行为吧！让你比较有印象啦！

例题：仿真伺服器重新安装后，假设服务器使用相同的 IP ，造成相同 IP 的服务器公钥不同，产生的问题与解决之道为何？答：利用前一小节讲过的方式，删除原有的系统公钥，重新启动 ssh 让你的公钥更新：

```
rm  /etc/ssh/ssh_host*
/etc/init.d/sshd restart
```

然后重新使用底下的方式来进行联机的动作：

```
[root@www ~]# ssh root@localhost
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @ &lt;==就告诉你可能有问题
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
a7:2e:58:51:9f:1b:02:64:56:ea:cb:9c:92:5e:79:f9.
Please contact your system administrator.
Add correct host key in /root/.ssh/known_hosts to get rid of this message.
Offending key in /root/.ssh/known_hosts:1 &lt;==冒号后面接的数字就是有问题数据行号
RSA host key for localhost has changed and you have requested strict checking.
Host key verification failed.
```

上述的表格出现的错误讯息中，特殊字体的地方在告诉你：/root/.ssh/known*hosts* 的第 *1* 行，里面的公钥与这次接收到的结果不同， 很可能被攻击了！那怎办？没关系啦！请你使用 *vim* 到 */root/.ssh/known_hosts* ，并将第 *1* 行*(*冒号 *:* 后面接的数字就是了*)* 删除，之后再重新 *ssh* 过，那系统又会重新问你要不要加上公钥啰！就这么简单！ ^^

- 模拟 FTP 的文件传输方式： sftp

ssh 是登入远程服务器进行工作，那如果你只是想要从远程服务器下载或上传档案呢？ 那就不是使用 ssh 啦，而必须要使用 sftp 或 scp。这两个指令也都是使用 ssh 的通道 (port 22)，只是模拟成 FTP 与复制的动作而已。我们先谈谈 sftp ，这个指令的用法与 ssh 很相似，只是 ssh 是用在登入而 sftp 在上传/下载文件而已。

```
[root@www ~]# sftp student@localhost
Connecting to localhost...
student@localhost's password: &lt;== 这里请输入密码啊！
sftp&gt; exit  &lt;== 这里就是在等待你输入 ftp 相关指令的地方了！
```

进入到 sftp 之后，那就跟在一般 FTP 模式下的操作方法没有两样了！底下我们就来谈一谈，sftp 这个接口下的使用指令吧！

针对远方服务器主机 **(Server)** 之行为	
变换目录到 /etc/test 或其他目录	cd /etc/test cd PATH
列出目前所在目录下的文件名	ls dir
建立目录	mkdir directory
删除目录	rmdir directory
显示目前所在的目录	pwd

更改档案或目录群组	chgrp groupname PATH
更改档案或目录拥有者	chown username PATH
更改档案或目录的权限	chmod 644 PATH 其中，644 与权限有关！回去看基础篇！
建立连结档	ln oldname newname
删除档案或目录	rm PATH
更改档案或目录名称	rename oldname newname
离开远程主机	exit (or) bye (or) quit
针对本机(Client) 之行为(都加上 l, L 的小写)	
变换目录到本机的 PATH 当中	lcd PATH
列出目前本机所在目录下的文件名	lls
在本机建立目录	lmkdir
显示目前所在的本机目录	lpwd
针对资料上传/下载的行为	
将档案由本机上传到远程主机	put [本机目录或档案] [远程] put [本机目录或档案] 如果是这种格式，则档案会放置到目前远程主机的目录下！
将档案由远程主机下载回来	get [远程主机目录或档案] [本机] get [远程主机目录或档案] 若是这种格式，则档案会放置在目前本机所在的目录当中！可以使用通配符，例如： get * get *.rpm 亦是可以的格式！

就整体而言， sftp 在 Linux 底下，如果不考虑图形接口，那么他已经可以取代 FTP 了呢！因为所有的功能都已经涵盖啦！因此，在不考虑到图形接口的 FTP 软件时，可以直接关掉 FTP 的服务，而改以 sftp-server 来提供 FTP 的服务吧！ ^_^

例题：假设 localhost 为远程服务器，且服务器上有 student 这个使用者。你想要 (1)将本机的 /etc/hosts 上传到 student 家目录，并 (2)将 student 的 .bashrc 复制到本机的 /tmp 底下，该如何透过 sftp 达成？答：

```
[root@www ~]# sftp student@localhost
sftp&gt; lls /etc/hosts   &lt;==先看看本机有没有这个档案
/etc/hosts
sftp&gt; put /etc/hosts   &lt;==有的话，那就上传吧！
Uploading /etc/hosts to /home/student/hosts
/etc/hosts                        100%  243     0.2KB/s   00:00
sftp&gt; ls               &lt;==有没有上传成功？看远程目录下的文件名
hosts
sftp&gt; ls -a            &lt;==那有没有隐藏档呢？
.               ..              .bash_history    .bash_logout
.bash_profile   .bashrc         .mozilla         hosts
sftt&gt; lcd /tmp         &lt;==切换本机目录到 /tmp
sftp&gt; lpwd             &lt;==只是进行确认而已！
Local working directory: /tmp
sftp&gt; get .bashrc      &lt;==没问题就下载吧！
Fetching /home/student/.bashrc to .bashrc
/home/student/.bashrc             100%  124     0.1KB/s   00:00
sftp&gt; lls -a           &lt;==看本地端档案档名
.        .font-unix   keyring-rNd7qX  .X11-unix
..       .gdm_socket  lost+found      scim-panel-socket:0-root
.bashrc  .ICE-unix    mapping-root    .X0-lock
sftp&gt; exit             &lt;==离开吧！
```

如果你不喜欢使用文字接口进行 FTP 的传输，那么还可以透过图形接口来连接到 sftp-server 哩！ 你可以利用二十一章 FTP 服务器提到的 Filezilla 来进行联机的啦！ 如此一来，与服务器之间的文件传输就方便多了吧！

- 档案异地直接复制： scp

通常使用 sftp 是因为可能不知道服务器上面有什么档名的档案存在，如果已经知道服务器上的档案档名了， 那么最简单的文件传输则是透过 scp 这个指令喔！最简单的 scp 用法如下：

```
[root@www ~]# scp [-pr] [-l 速率] file  [账号@]主机:目录名 &lt;==上传
[root@www ~]# scp [-pr] [-l 速率] [账号@]主机:file  目录名 &lt;==下载
选项与参数：
-p  ：保留原本档案的权限数据；
-r  ：复制来源为目录时，可以复制整个目录 (含子目录)
-l  ：可以限制传输的速度，单位为 Kbits/s ，例如 [-l 800] 代表传输速限 100Kbytes/s

# 1\. 将本机的 /etc/hosts* 全部复制到 127.0.0.1 上面的 student 家目录内
[root@www ~]# scp /etc/hosts* student@127.0.0.1:~
student@127.0.0.1's password: &lt;==输入 student 密码
hosts                       100%  207         0.2KB/s   00:00
hosts.allow                 100%  161         0.2KB/s   00:00
hosts.deny                  100%  347         0.3KB/s   00:00
# 文件名显示                  进度  容量(bytes) 传输速度  剩余时间
# 你可以仔细看，出现的讯息有五个字段，意义如上所示。

# 2\. 将 127.0.0.1 这部远程主机的 /etc/bashrc 复制到本机的 /tmp 底下
[root@www ~]# scp student@127.0.0.1:/etc/bashrc /tmp
```

其实上传或下载的重点是那个冒号 (:) 啰！连接在冒号后面的就是远程主机的档案。 因此，如果冒号在前，代表的就是从远程主机下载下来，如果冒号在后，则代表本机数据上传啦！ 而如果想要复制目录的话，那么可以加上 -r 的选项！

例题：假设本机有个档案档名为 /root/dd_10mb_file ，这个档案有 10 MB 这么大。假设你想要上传到 127.0.0.1 的 /tmp 底下去， 而且你在 127.0.0.1 上面有 root 这个账号的使用权。但由于带宽很宝贵，因此你只想要花费 100Kbyes/s 的传输量给此一动作， 那该如何下达指令？答：由于预设不存在这个档案，因此我们得先使用 dd 来建立一个大档案：

```
dd if=/dev/zero of=/root/dd_10mb_file bs=1M count=10
```

建立妥当之后，由于是上传数据，观察 -l 的选项中，那个速率用的是 bit ，转成容量的 bytes 需要乘上 8 倍，因此指令就要这样下达：

```
scp -l 800 /root/dd_10mb_file root@127.0.0.1:/tmp
```

11.2.4 ssh 客户端联机程序 - Windows 用户

与 Linux 不同的是，预设的 Windows 并没有 ssh 的客户端程序，因此所有的程序都得要下载其他第三方软件才行。 常见的软件主要有 pietty, psftp 及 filezilla 等。底下就让我们来谈谈这几个软件吧。

- 直接联机的 pietty

在 Linux 底下想要连接 SSH 服务器，可以直接利用 ssh 这个指令，在 Windows 操作系统底下就得要使用 pietty 或 putty 这两个玩意儿，这两者的下载点请参考 (注2)：

- putty 官方网站：http://www.chiark.greenend.org.uk/~sgtatham/putty/
- pietty 官方网站：http://www.csie.ntu.edu.tw/~piaip/pietty/

在 putty 的官方网站上有很多的软件可以使用的，包括 putty/pscp/psftp 等等。他们分别对应了 ssh/scp/sftp 这三个指令就是了。而鸟哥爱用的 pietty 则是台湾的林弘德先生根据 putty 所改版而成的。由于 pietty 除了完整的兼容于 putty 之外，还提供了选单与较为完整的文字编码，实在很好用呢，所以底下鸟哥就以 pietty 来作为介绍啰。在你下载 pietty 完成后，双击该档案，就会出现如下的画面啰：

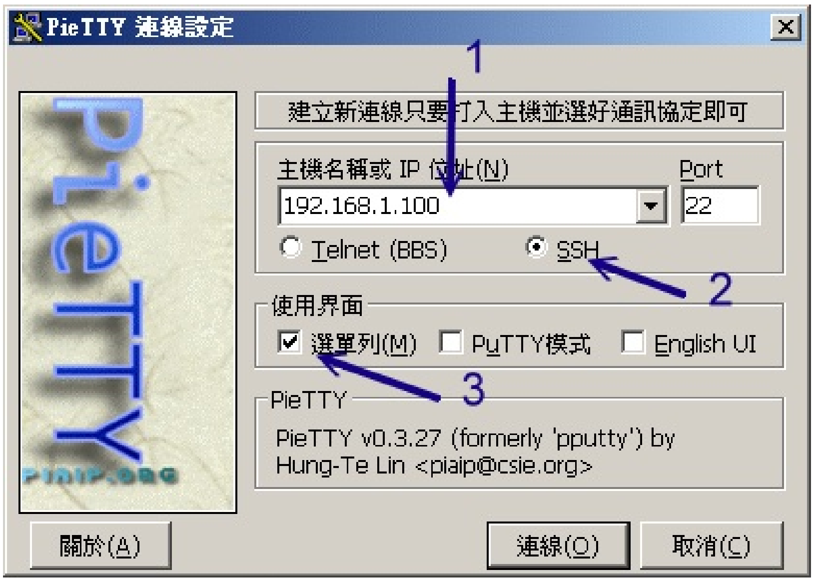


图 11.2-3、pietty 的启动屏幕示意图

在上图中箭头为 1 的地方请填写相关的主机名或者是 IP ，箭头 2 当然务必选择 SSH 那一项，至于箭头 3 的地方，鸟哥比较喜欢选单出现的样式，因为可以直接修改一些 pietty 的环境设定值，所以鸟哥是选择选单啦！ 若没有问题，按下『联机』后，就会出现如下等待登入与输入账/密数据的画面：



图 11.2-4、pietty 的登入与使用画面示意图

这个图标会让你以为是在主机前面工作吧！而且上头还有选单可以随时调整类似字形、字体、字符编码等等的重要环境参数。 尤其是字符编码的问题，有时候你会发现开启档案时，竟然画面当中会有乱码而不是正常的中文显示， 那就是编码的问题。要解决这个问题时，你必须要牢记下面的三个跟语系编码有关的数据要相同才行：

- 文本文件本身在存档时所挑选的语系；
- Linux 程序 (如 bash 软件) 本身所使用的语系 (可用 LANG 变量调整)；
- pietty 所使用的语系。

我们知道 Linux 本身的编码可以透过 LANG 这个变量来调整，那该如何调整 pietty 的中文编码呢？你可以透过图 11.2-4 选单列当中的『选项』来处理，如下所示：

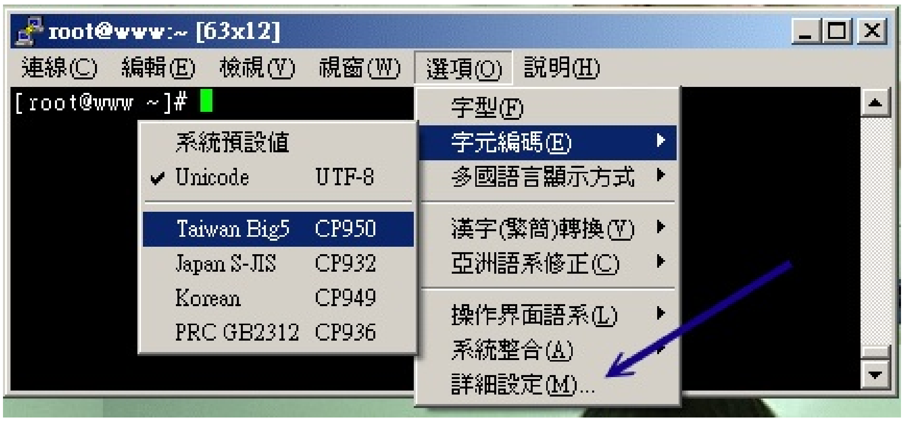


图 11.2-5、调整 pietty 的语系编码方式 (与中文较相关)

在『选项』的『字符编码』里面可以挑选 big5 (cp950) 或者是 unicode (utf8) 的中文编码，让它符合你的 Linux 与档案所储存的数据格式，那中文字就 OK 的啦！ ^_^！如果想要作更细部的设定时，可以选择图 11.2-5 上头最底下的那个『详细设定』项目， 就会出现如下图示。其中更为重要的是『键盘右侧的数字键想要生效』时， 可以按照下图的指示来启动数字键的功能：

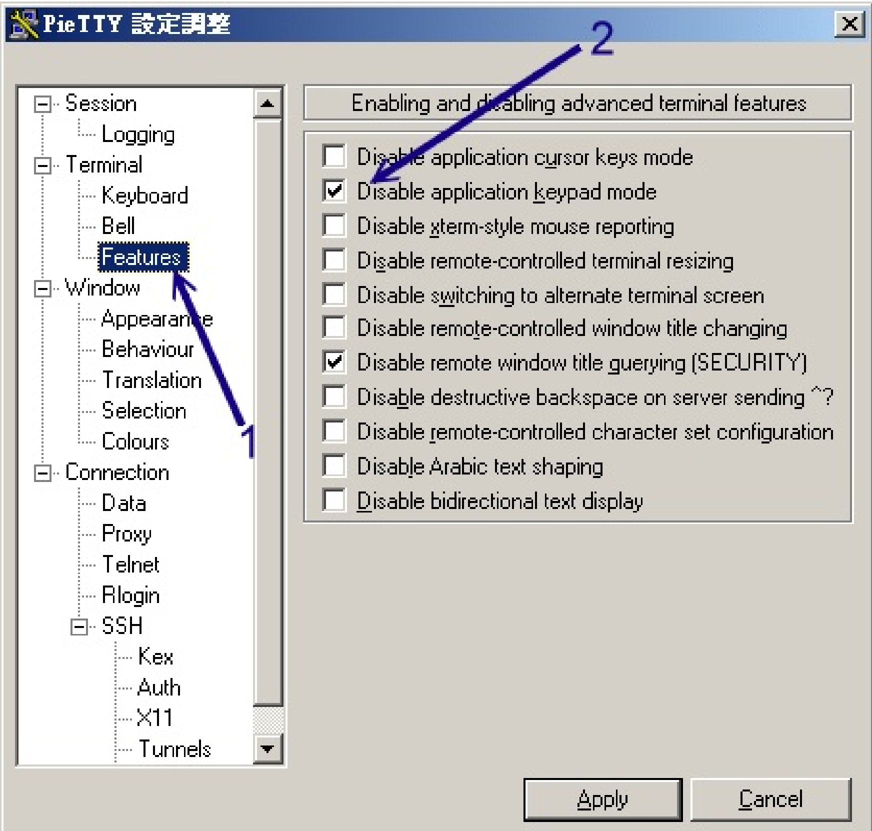


图 11.2-6、pietty 软件环境详细设定，与键盘右侧数字键相关者

将上图中箭头 2 所指的那个项目勾选起来且按下『Apply』之后，你键盘右侧的数字键才能够正常的使用呢，否则按右侧数字键会是乱码啦。 再来，你可以调整 pietty 滚动条的记忆行数，这样当数据太多时，你依旧可以调整滚动条来查阅之前的数据。设定的方法如下：

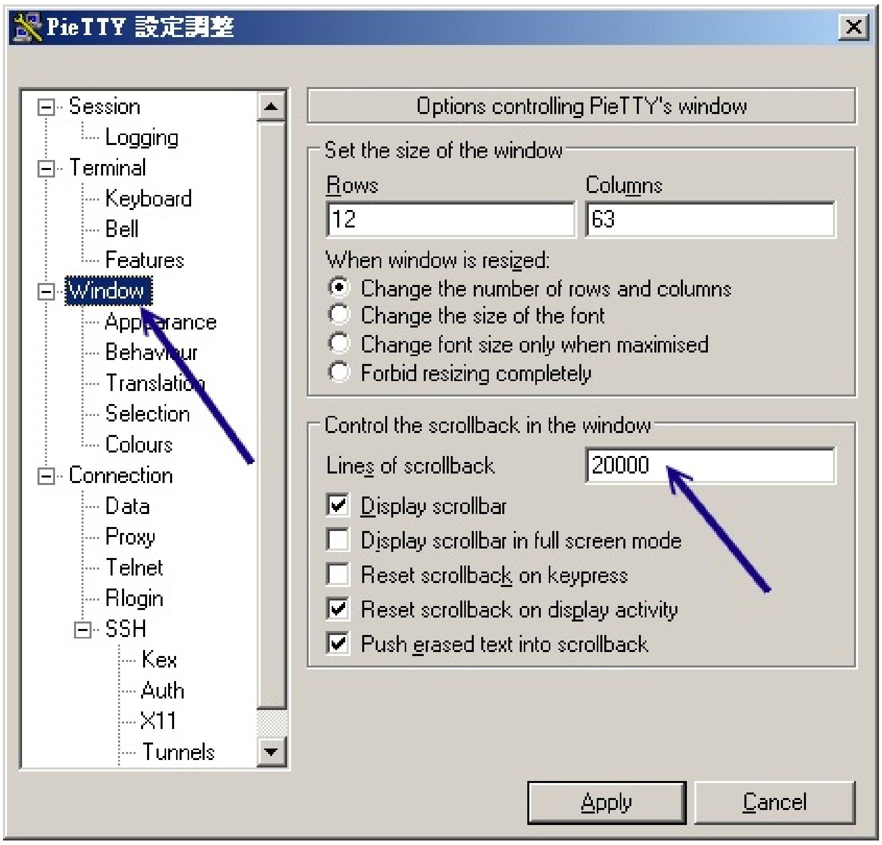


图 11.2-7、调整画面可以记忆的行数，可让用户回去看较多之前的画面

调整完这些常用的数据后，再来这是最重要的：『你要以哪一个版本的 SSH 算法登入？』前面说过，我们预设是以 version2 来登入的，所以这里我们可以调整为 2 那个项目！这样每次登入都会以 version 2 的模式登入主机了！

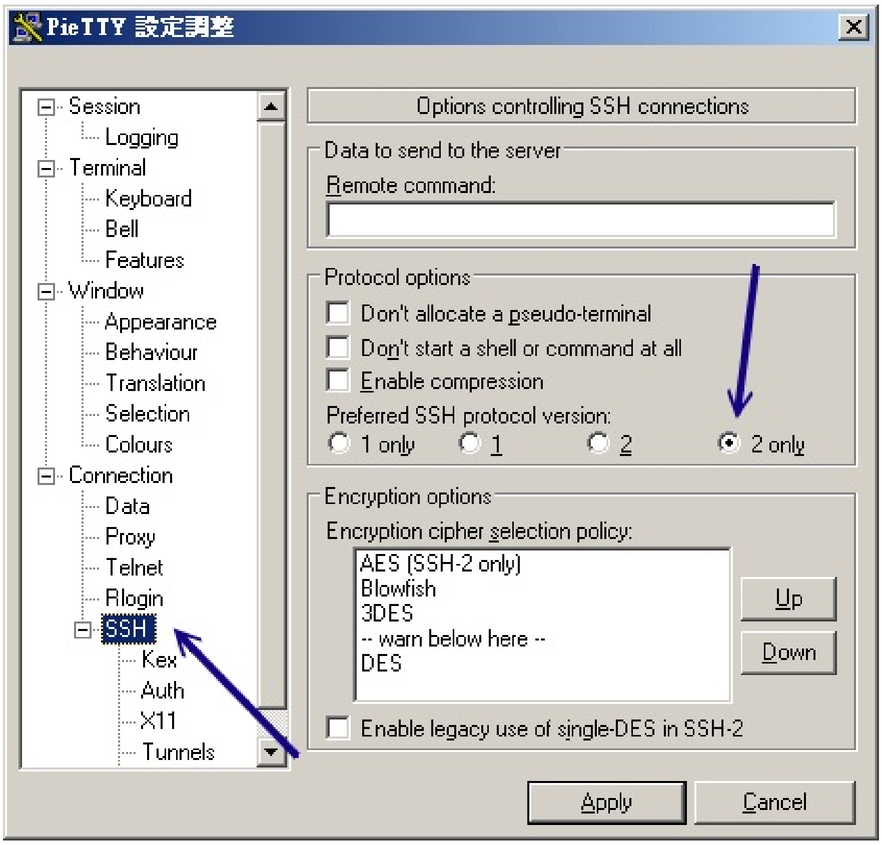


图 11.2-8、设定登入服务器时使用的 ssh 算法版本

整个 pietty 的使用与相关设定流程就是这样！如此一来，你就可以在 Windows 上面以 SSH 的协议，登入远程的 Linux 主机噜！粉方便吧！ ^_^ ！如果想要中文支持的话，目前 pietty 已经支持中文啦！你可以输入中文喔！不过需要修改一下字符集， 选择图 11.2-5 『选项』内的『字型』就会出现如下图示：

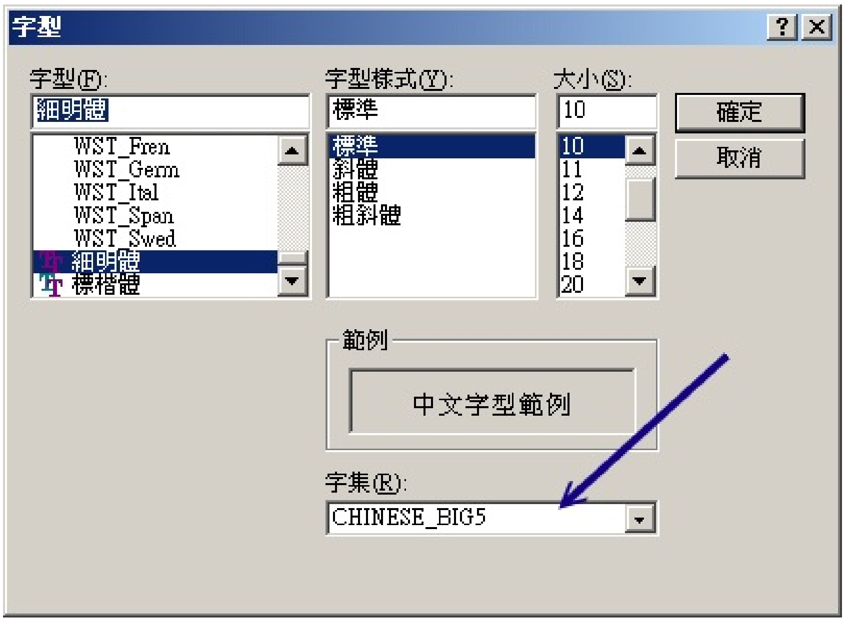


图 11.2-9、选择中文的字形与编码

将(1)字型设定为细明体、(2)字集设定为『Big5』，如此一来，你的 pietty 就支持中文的输入啰！

那么上面我们作的这些设定值都记录在哪里啊？呵呵！都记录在 Windows 的登录文件当中啊！你可以在 Windows 的系统当中，在『开始』--> 『执行』后，出现的框框内输入『regedit』，之后会出现一个大窗口。请在左边的画面当中选择『 HKEY*CURRENT_USER --> Software --> SimonTatham --> PuTTY --> Sessions*』， 就可以看到你的设定值啰！ ^^！这样，也就可以储存你的设定值啰～

- 使用 sftp-server 的功能： psftp

在 putty 的官方网站上也提供 psftp 这支程序。这一支程序的重点则在使用 sftp-server。使用的方式可以直接点选 psftp 这个档案，让他直接启动，则会出现下面的图样：

```
psftp: no hostname specified; use "open host.name" to connect
psftp&gt;
```

这个时候可以填入你要连接上去的主机名，例如我的区域内网络 192.168.100.254 这部主机：

```
psftp: no hostname specified; use "open host.name" to connect
psftp&gt; open 192.168.100.254
login as: root
root@192.168.100.254's password:
Remote working directory is /root
psftp&gt; &lt;== 这里就在等待你输入 FTP 的指令了！
```

呵呵！这样就登入主机啦！很简单吧！然后其他的使用方式跟前面提到的 sftp 一样哩！加油的使用吧！

- 图形化接口的 sftp 客户端软件： Filezilla

SSH 所提供的 sftp 功能只能利用纯文本接口的 psftp 来联机吗？有没有图形接口的软件呢？呵呵！当然有！ 那就是非常有用的 Filezilla 啰！Filezilla 是图形接口的一个 FTP 客户端软件，使用上非常的方便， 至于详细的安装与使用流程请参考第二十一章 vsftpd 的说明喔！

11.2.5 sshd 服务器细部设定

基本上，所有的 sshd 服务器详细设定都放在 /etc/ssh/sshd_config 里面！不过，每个 Linux distribution 的预设设定都不太相同，所以我们有必要来了解一下整个设定值的意义为何才好！ 同时请注意，在预设的档案内，只要是预设有出现且被批注的设定值 (设定值前面加 #)，即为『默认值！』，你可以依据它来修改的哩。

```
[root@www ~]# vim /etc/ssh/sshd_config
# 1\. 关于 SSH Server 的整体设定，包含使用的 port 啦，以及使用的密码演算方式
# Port 22
# SSH 预设使用 22 这个port，也可以使用多个port，即重复使用 port 这个设定项目！
# 例如想要开放 sshd 在 22 与 443 ，则多加一行内容为：『 Port 443 』
# 然后重新启动 sshd 这样就好了！不过，不建议修改 port number 啦！
```

```
Protocol 2
# 选择的 SSH 协议版本，可以是 1 也可以是 2 ，CentOS 5.x 预设是仅支援 V2。
# 如果想要支持旧版 V1 ，就得要使用『 Protocol 2,1 』才行。

# ListenAddress 0.0.0.0
# 监听的主机适配器！举个例子来说，如果你有两个 IP，分别是 192.168.1.100 及
# 192.168.100.254，假设你只想要让 192.168.1.100 可以监听 sshd ，那就这样写：
# 『 ListenAddress 192.168.1.100 』默认值是监听所有接口的 SSH 要求

# PidFile /var/run/sshd.pid
# 可以放置 SSHD 这个 PID 的档案！上述为默认值

# LoginGraceTime 2m
# 当使用者连上 SSH server 之后，会出现输入密码的画面，在该画面中，
# 在多久时间内没有成功连上 SSH server 就强迫断线！若无单位则默认时间为秒！

# Compression delayed
# 指定何时开始使用压缩数据模式进行传输。有 yes, no 与登入后才将数据压缩 (delayed)

# 2\. 说明主机的 Private Key 放置的档案，预设使用下面的档案即可！
# HostKey /etc/ssh/ssh_host_key        # SSH version 1 使用的私钥
# HostKey /etc/ssh/ssh_host_rsa_key    # SSH version 2 使用的 RSA 私钥
# HostKey /etc/ssh/ssh_host_dsa_key    # SSH version 2 使用的 DSA 私钥
# 还记得我们在主机的 SSH 联机流程里面谈到的，这里就是 Host Key ～

# 3\. 关于登录文件的讯息数据放置与 daemon 的名称！
SyslogFacility AUTHPRIV
# 当有人使用 SSH 登入系统的时候，SSH 会记录信息，这个信息要记录在什么 daemon name
# 底下？预设是以 AUTH 来设定的，即是 /var/log/secure 里面！什么？忘记了！
# 回到 [Linux 基础](http://linux.vbird.org/linux_basic/)去翻一下。其他可用的 daemon name 为：DA
# LOCAL0,LOCAL1,LOCAL2,LOCAL3,LOCAL4,LOCAL5,

# LogLevel INFO
# 登录记录的等级！嘿嘿！任何讯息！同样的，忘记了就回去参考！

# 4\. 安全设定项目！极重要！
# 4.1 登入设定部分
# PermitRootLogin yes
# 是否允许 root 登入！预设是允许的，但是建议设定成 no！

# StrictModes yes
# 是否让 sshd 去检查用户家目录或相关档案的权限数据，
# 这是为了担心使用者将某些重要档案的权限设错，可能会导致一些问题所致。
# 例如使用者的 ~.ssh/ 权限设错时，某些特殊情况下会不许用户登入

# PubkeyAuthentication yes
# AuthorizedKeysFile      .ssh/authorized_keys
# 是否允许用户自行使用成对的密钥系统进行登入行为，仅针对 version 2。
# 至于自制的公钥数据就放置于用户家目录下的 .ssh/authorized_keys 内

PasswordAuthentication yes
# 密码验证当然是需要的！所以这里写 yes 啰！

# PermitEmptyPasswords no
# 若上面那一项如果设定为 yes 的话，这一项就最好设定为 no ，
# 这个项目在是否允许以空的密码登入！当然不许！

# 4.2 认证部分
# RhostsAuthentication no
# 本机系统不使用 .rhosts，因为仅使用 .rhosts太不安全了，所以这里一定要设定为 no

# IgnoreRhosts yes
# 是否取消使用 ~/.ssh/.rhosts 来做为认证！当然是！

# RhostsRSAAuthentication no #
# 这个选项是专门给 version 1 用的，使用 rhosts 档案在 /etc/hosts.equiv
# 配合 RSA 演算方式来进行认证！不要使用啊！

# HostbasedAuthentication no
# 这个项目与上面的项目类似，不过是给 version 2 使用的！

# IgnoreUserKnownHosts no
```

```
# 是否忽略家目录内的 ~/.ssh/known_hosts 这个档案所记录的主机内容？
# 当然不要忽略，所以这里就是 no 啦！

ChallengeResponseAuthentication no
# 允许任何的密码认证！所以，任何 login.conf 规定的认证方式，均可适用！
# 但目前我们比较喜欢使用 PAM 模块帮忙管理认证，因此这个选项可以设定为 no 喔！

UsePAM yes
# 利用 PAM 管理使用者认证有很多好处，可以记录与管理。
# 所以这里我们建议你使用 UsePAM 且 ChallengeResponseAuthentication 设定为 no

# 4.3 与 Kerberos 有关的参数设定！因为我们没有 Kerberos 主机，所以底下不用设定！
# KerberosAuthentication no
# KerberosOrLocalPasswd yes
# KerberosTicketCleanup yes
# KerberosTgtPassing no

# 4.4 底下是有关在 X-Window 底下使用的相关设定！
X11Forwarding yes
# X11DisplayOffset 10
# X11UseLocalhost yes
# 比较重要的是 X11Forwarding 项目，他可以让窗口的数据透过 ssh 信道来传送喔！
# 在本章后面比较进阶的 ssh 使用方法中会谈到。

# 4.5 登入后的项目：
# PrintMotd yes
# 登入后是否显示出一些信息呢？例如上次登入的时间、地点等等，预设是 yes
# 亦即是打印出 /etc/motd 这个档案的内容。但是，如果为了安全，可以考虑改为 no ！

# PrintLastLog yes
# 显示上次登入的信息！可以啊！预设也是 yes ！

# TCPKeepAlive yes
# 当达成联机后，服务器会一直传送 TCP 封包给客户端藉以判断对方式否一直存在联机。
# 不过，如果联机时中间的路由器暂时停止服务几秒钟，也会让联机中断喔！
# 在这个情况下，任何一端死掉后，SSH可以立刻知道！而不会有僵尸程序的发生！
# 但如果你的网络或路由器常常不稳定，那么可以设定为 no 的啦！

UsePrivilegeSeparation yes
# 是否权限较低的程序来提供用户操作。我们知道 sshd 启动在 port 22 ，
# 因此启动的程序是属于 root 的身份。那么当 student 登入后，这个设定值
# 会让 sshd 产生一个属于 sutdent 的 sshd 程序来使用，对系统较安全

MaxStartups 10
# 同时允许几个尚未登入的联机画面？当我们连上 SSH ，但是尚未输入密码时，
# 这个时候就是我们所谓的联机画面啦！在这个联机画面中，为了保护主机，
# 所以需要设定最大值，预设最多十个联机画面，而已经建立联机的不计算在这十个当中

# 4.6 关于用户抵挡的设定项目：
DenyUsers *
# 设定受抵挡的使用者名称，如果是全部的使用者，那就是全部挡吧！
# 若是部分使用者，可以将该账号填入！例如下列！
DenyUsers test

DenyGroups test
# 与 DenyUsers 相同！仅抵挡几个群组而已！

# 5\. 关于 SFTP 服务与其他的设定项目！
Subsystem       sftp    /usr/lib/ssh/sftp-server
# UseDNS yes
# 一般来说，为了要判断客户端来源是正常合法的，因此会使用 DNS 去反查客户端的主机名
# 不过如果是在内网互连，这项目设定为 no 会让联机达成速度比较快。
```

基本上，CentOS 预设的 sshd 服务已经算是挺安全的了，不过还不够！建议你 (1)将 root 的登入权限取消；(2)将 ssh 版本设定为 2 。其他的设定值就请你依照自己的喜好来设定了。通常不建议进行随便修改啦！另外，如果你修改过上面这个档案(/etc/ssh/sshd_config)，那么

就必需要重新启动一次 sshd 这个 daemon 才行！亦即是：

- /etc/init.d/sshd restart

11.2.6 制作不用密码可立即登入的 ssh 用户

你或许已经想到了，既然 ssh 可以使用 scp 来进行网络复制的话，那么我能不能将 scp 的指令放置于 crontab 服务中， 让我们的系统透过 scp 直接在背景底下自行定期的进行网络复制与备份呢？抱歉，答案是：『预设状况下不允许此动作』的！ 为甚么呢？因为预设状况下，你必须要透过远程登录，与 scp 互动的输入密码才行啊！但 crontab 又不会让你有终端接口输入密码， 所以该程序就会一直卡住而无法在 crontab 内执行成功喔！ 那怎办？我们要放弃这个好用的网络复制工具吗？当然不是啦！我们可以透过密钥认证系统来处理的！

既然 SSH 可以使用密钥系统来比对数据，并且提供用户数据的加密功能，那么可不可能利用这个 Key 就提供用户自己进入主机，而不需要输入密码呢？呵呵！好主意！我们可以将 Client 产生的 Key 给他拷贝到 Server 当中，所以， 以后 Client 登入 Server 时，由于两者在 SSH 要联机的讯号传递中，就已经比对过 Key 了， 因此，可以立即进入数据传输接口中，而不需要再输入密码呢！在实作上的步骤可以是：

1. 客户端建立两把钥匙：想一想，在密钥系统中，是公钥比较重要还是私钥比较重要？ 当然是私钥比较重要！因此私钥才是解密的关键啊！所以啰，这两把钥匙当然得在发起联机的客户端建置才对。利用的指令为 ssh-keygen 这个命令；

2. 客户端放置好私钥档案：将 Private Key 放在 Client 上面的家目录，亦即 $HOME/.ssh/ ， 并且得要注意权限喔！

3. 将公钥放置服务器端的正确目录与文件名去：最后，将那把 Public Key 放在任何一个你想要用来登入的服务器端的某 User 的家目录内之 .ssh/ 里面的认证档案即可完成整个程序。

说是好像很困难的样子，其实步骤眞的很简单，我们依序来进行作业好了！假设前提如下，该进行的步骤则如下图：

- Server 部分为 www.centos.vbird 这部 192.168.100.254 的主机，欲使用的账号为 dmtsai ；
- Client 部分为 clientlinux.centos.vbird 这部 192.168.100.10 的 vbirdtsai 这个账号， 该账号要用来登入 192.168.100.254 这部主机的 dmtsai 账号。



图 11.2-10、制作不需要密码的 ssh 账号基本流程

- 1. 客户端建立两把钥匙：

建立的方法很简单，在 clientlinux.centos.vbird 这部主机上面以 vbirdtsai 的身份来建立两把钥匙即可。 不过，需要注意的是，我们有多种密码算法，如果不指定特殊的算法，则默认以 RSA 算法来处理：

```
[vbirdtsai@clientlinux ~]$ ssh-keygen [-t rsa&#124;dsa] &lt;==可选 rsa 或 dsa
[vbirdtsai@clientlinux ~]$ ssh-keygen  &lt;==用预设的方法建立密钥
Generating public/private rsa key pair.
Enter file in which to save the key (/home/vbirdtsai/.ssh/id_rsa): &lt;==按 enter
Created directory '/home/vbirdtsai/.ssh'. &lt;==此目录若不存在则会主动建立
Enter passphrase (empty for no passphrase): &lt;==按 Enter 不给密码
Enter same passphrase again: &lt;==再输入一次 Enter 吧！
Your identification has been saved in /home/vbirdtsai/.ssh/id_rsa. &lt;==私钥档
Your public key has been saved in /home/vbirdtsai/.ssh/id_rsa.pub. &lt;==公钥档
The key fingerprint is:
0f:d3:e7:1a:1c:bd:5c:03:f1:19:f1:22:df:9b:cc:08 vbirdtsai@clientlinux.centos.vbird

[vbirdtsai@clientlinux ~]$ ls -ld ~/.ssh; ls -l ~/.ssh
drwx------. 2 vbirdtsai vbirdtsai 4096 2011-07-25 12:58 /home/vbirdtsai/.ssh
-rw-------. 1 vbirdtsai vbirdtsai 1675 2011-07-25 12:58 id_rsa      &lt;==私钥档
-rw-r--r--. 1 vbirdtsai vbirdtsai  416 2011-07-25 12:58 id_rsa.pub  &lt;==公钥档
```

请注意上面喔，我的身份是 vbirdtsai ，所以当我执行 ssh-keygen 时，才会在我的家目录底下的 .ssh/ 这个目录里面产生所需要的两把 Keys ，分别是私钥 (id_rsa) 与公钥 (id_rsa.pub)。 ~/.ssh/ 目录必须要是 700 的权限才行！另外一个要特别注意的就是那个 id_rsa 的档案权限啦！他必须要是 -rw------- 且属于 vbirdtsai 自己才行！否则在未来密钥比对的过程当中，可能会被判定为危险而无法成功的以公私钥成对档案的机制来达成联机喔。 其实，建立私钥后预设的权限与文件名放置位置都是正确的，你只要检查过没问题即可。

- 2. 将公钥档案数据上传到服务器上：

因为我们要登入 www.centos.vbird 是以 dmtsai 的身份，因此我们就得要将上个步骤建立的公钥 (id_rsa.pub) 上传到服务器上的 dmtsai 用户才行。那如何上传呢？最简单的方法当然就是使用 scp 嘛！

```
[vbirdtsai@clientlinux ~]$ scp ~/.ssh/id_rsa.pub dmtsai@192.168.100.254:~
# 上传到 dmtsai 的家目录底下即可。
```

- 3. 将公钥放置服务器端的正确目录与文件名：

还记得 sshd_config 里面谈到的 AuthorizedKeysFile 这个设定值吧？该设定值就是在指定公钥数据应该要放置的文件名啰！所以，我们必须要到服务器端的 dmtsai 这个用户身份下， 将刚刚上传的 id_rsa.pub 数据附加到 authorized_keys 这个档案内才行。作法有点像这样：

```
# 1\. 建立 ~/.ssh 档案，注意权限需要为 700 喔！
[dmtsai@www ~]$ ls -ld .ssh
ls: .ssh: 没有此一档案或目录
# 由于可能是新建的用户，因此这个目录不存在。不存在才作底下建立目录的行为

[dmtsai@www ~]$ mkdir .ssh; chmod 700 .ssh
[dmtsai@www ~]$ ls -ld .ssh
drwx------. 2 dmtsai dmtsai 4096 Jul 25 13:06 .ssh
# 权限设定中，务必是 700 且属于使用者本人的账号与群组才行！

# 2\. 将公钥档案内的数据使用 cat 转存到 authorized_keys 内
[dmtsai@www ~]$ ls -l *pub
-rw-r--r--. 1 dmtsai dmtsai 416 Jul 25 13:05 id_rsa.pub &lt;==确实有存在

[dmtsai@www ~]$ cat id_rsa.pub &gt;&gt; .ssh/authorized_keys
[dmtsai@www ~]$ chmod 644 .ssh/authorized_keys
[dmtsai@www ~]$ ls -l .ssh
-rw-r--r--. 1 dmtsai dmtsai 416 Jul 25 13:07 authorized_keys
# 这个档案的权限设定中，就得要是 644 才可以！不可以搞混了！
```

这样就搞定密钥系统啰！以后你从 clientlinux.centos.vbird 的 vbirdtsai 登入到 www.centos.vbird 的 dmtsai 用户时， 就不需要任何的密码啰！举例来说，你可以这样测试看看啰：

例题：透过上述的案例练习成功后，请在 clientlinux 的 vbirdtsai 身份中，将系统的 /etc/hosts* 档案复制给 www.centos.vbird 的 dmtsai 用户的家目录。答：

```
[vbirdtsai@clientlinux ~]$ scp /etc/hosts* dmtsai@192.168.100.254:~
hosts                                         100%  187     0.2KB/s   00:00
hosts.allow                                   100%  161     0.2KB/s   00:00
hosts.deny                                    100%  347     0.3KB/s   00:00
# 你会发现，原本会出现的那个密码提示数据不会出现了喔！

[vbirdtsai@clientlinux ~]$ ssh dmtsai@192.168.100.254 "ls -l"
-rw-r--r--. 1 dmtsai dmtsai 196 2011-07-25 13:09 hosts
-rw-r--r--. 1 dmtsai dmtsai 370 2011-07-25 13:09 hosts.allow
-rw-r--r--. 1 dmtsai dmtsai 460 2011-07-25 13:09 hosts.deny
-rw-r--r--. 1 dmtsai dmtsai 416 2011-07-25 13:05 id_rsa.pub
# 确实有复制到对方去了！有显示出正确的远程数据哩！
```

很简单的步骤吧！这样一来，使用 ssh 相关的客户端指令就可以不需密码的手续了！无论如何，在建立密钥系统的步骤中你要记得的是：

- Client 必须制作出 Public & Private 这两把 keys，且 Private 需放到 ~/.ssh/ 内；
- Server 必须要有 Public Key ，且放置到用户家目录下的 ~/.ssh/authorized_keys，同时目录的权限 (.ssh/) 必须是 700 而档案权限则必须为 644 ，同时档案的拥有者与群组都必须与该账号吻合才行。

未来，当你还想要登入其他的主机时，只要将你的 public key (就是 id_rsa.pub 这个档案) 给他 copy 到其他主机上面去，并且新增到某账号的 ~/.ssh/authorized_keys 这个档案中！哈哈！成功！

11.2.7 简易安全设定

老实说，大家都被『SSH 是个安全的服务』所欺骗了！其实 sshd 并不怎么安全的！翻开 openssh 的过去历史来看，确实有很多人是利用 ssh 的程序漏洞来取得远程主机 root 的权限，进一步黑掉对方的主机！所以这玩意儿说实话，也不是很安全的啦！

sshd 之所谓的『安全』其实指的是『 sshd 的数据是加密过的，所以他的数据在 Internet 上面传递时是比较安全的。至于 sshd 这个服务本身就不是那样安全了！所以说：『非必要，不要将 sshd 对 Internet 开放可登入的权限，尽量局限在几个小范围内的 IP 或主机名即可！这很重要的喔！

好了，那么关于安全的设定方面，有没有什么值得注意的呢？当然是有啦！我们可以先建议几个项目吧！分别可以由底下这三方面来进行：

- 服务器软件本身的设定强化：/etc/ssh/sshd_config
- TCP wrapper 的使用：/etc/hosts.allow, /etc/hosts.deny
- iptables 的使用： iptables.rule, iptables.allow

- 服务器软件本身的设定强化：/etc/ssh/sshd_config

一般而言，这个档案的默认项目就已经很完备了！所以，事实上是不太需要更动他的！ 但是，如果你有些使用者方面的顾虑，那么可以这样修正一些问题呢！

- 禁止 root 这个账号使用 sshd 的服务；
- 禁止 nossh 这个群组的用户使用 sshd 的服务；
- 禁止 testssh 这个用户使用 sshd 的服务；

除了上述的账号之外，其他的用户则可以正常的使用系统。现在鸟哥假设你的系统里面已经有 sshnot1, sshnot2, sshnot3 加入 nossh 群组， 同时系统还有 testssh, student 等账号。相关的账号处理请自行参考基础篇来设定，底下仅是列出观察的重点：

```
# 1\. 先观察一下所需要的账号是否存在呢？
[root@www ~]# for user in sshnot1 sshnot2 sshnot3 testssh student; do \
&gt; id $user &#124; cut -d ' ' -f1-3 ; done
uid=507(sshnot1) gid=509(sshnot1) groups=509(sshnot1),508(nossh)
uid=508(sshnot2) gid=510(sshnot2) groups=510(sshnot2),508(nossh)
uid=509(sshnot3) gid=511(sshnot3) groups=511(sshnot3),508(nossh)
uid=511(testssh) gid=513(testssh) groups=513(testssh)
uid=505(student) gid=506(student) groups=506(student)
# 若上述账号并不存在你的系统，请自己建置出来！UID/GID 与鸟哥的不同也没关系！

# 2\. 修改 sshd_config 并且重新启动 sshd 吧！
[root@www ~]# vim /etc/ssh/sshd_config
PermitRootLogin no  &lt;==约在第 39 行，请拿掉批注且修改成这样
DenyGroups  nossh   &lt;==底下这两行可以加在档案的最后面
DenyUsers   testssh

[root@www ~]# /etc/init.d/sshd restart

# 3\. 测试与观察相关的账号登入情况吧！
[root@www ~]# ssh root@localhost  &lt;==并请输入正确的密码
[root@www ~]# tail /var/log/secure
Jul 25 13:14:05 www sshd[2039]: pam_unix(sshd:auth): authentication failure;
logname= uid=0 euid=0 tty=ssh ruser= rhost=localhost  user=root
# 你会发现出现这个错误讯息，而不是密码输入错误而已。

[root@www ~]# ssh sshnot1@localhost  &lt;==并请输入正确的密码
[root@www ~]# tail /var/log/secure
Jul 25 13:15:53 www sshd[2061]: User sshnot1 from localhost not allowed because
a group is listed in DenyGroups

[root@www ~]# ssh testssh@localhost  &lt;==并请输入正确的密码
[root@www ~]# tail /var/log/secure
Jul 25 13:17:16 www sshd[2074]: User testssh from localhost not allowed
because listed in DenyUsers
```

从上面的结果来看，你就会发现到，不同的登入账号会产生不一样的登录档结果。因此，当你老是无法顺利使用 ssh 登入某一部主机时，记得到该服务器上去检查看看登录档，说不定就会顺利的让你解决问题啰！在我们的测试机上面，请还是放行 root 的登入喔！

- /etc/hosts.allow 及 /etc/hosts.deny

举例来说，你的 sshd 只想让本机以及区网内的主机来源能够登入的话，那就这样作：

```
[root@www ~]# vim /etc/hosts.allow
sshd: 127.0.0.1 192.168.1.0/255.255.255.0 192.168.100.0/255.255.255.0

[root@www ~]# vim /etc/hosts.deny
sshd : ALL
```

- iptables 封包过滤防火墙

多几层保护也很好的！所以也可以使用 iptables 喔！ 参考：第九章、防火墙与 NAT 服务器内的实际脚本程序，你应该在 iptables.rule 内将 port 22 的放行功能取消，然后再到 iptables.allow 里面新增这行：

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.allow
iptables -A INPUT -i $EXTIF -s 192.168.1.0/24 -p tcp --dport 22 -j ACCEPT
iptables -A INPUT -i $EXTIF -s 192.168.100.0/24 -p tcp --dport 22 -j ACCEPT

[root@www ~]# /usr/local/virus/iptables/iptables.rule
```

上述的方法处理完毕后，如果你还是一部测试机，那么记得要将设定值还原回来吻！最后，『鸟哥呼吁大家，不要开放 SSH 的登入权限给所有 Internet 上面的主机～』 这很重要喔～因为如果对方可以 ssh 进入你的主机，那么...太危险了～

11.3 最原始图形接口： Xdmcp 服务的启用

考虑一个情况，如果你的 Linux 主机主要是用来作为图形处理时，而且同时有多人需要用到那个功能， 那么一部 Linux 是否一次仅能提供一个人处理那个软件呢？嘿嘿！那可不一定喔！因为 Linux 有相当优秀的 X Window System 啊！现在就来谈谈第一个图形接口的远程联机服务器吧！

11.3.1 X Window 的 Server/Client 架构与各组件

由于我们 Linux 使用的图形接口是所谓的 X-Window System 的东西，这玩意儿是能够跨平台的，目前在 Linux 上头开发的图形接口软件，几乎都是使用这个 X 的架构来处理，所以啰，你就不能够不知道 X Window 啦！ 我们在基础篇第三版的二十四章已经讲过 X Window 啦，因此这里只会作个简单的介绍，以方便大家来了解为何我们的软件是这么安装与设定喔！

X Window System 在运作的过程中，又因控制的数据不同而分为 X Server 与 X Client 两种程序，虽然说是 X Server/Client ， 但是他的作用却与网络主机的 Server/Client 架构大异其趣喔～先来说说 X Server/Client 这两种程序所负责的任务先：

- X Server： 这组程序主要负责的是屏幕画面的绘制与显示。 X Server 可以接收来自 X client 的数据，将这些数据绘制呈现为图面在屏幕上。 此外，我们移动鼠标、点击数据、由键盘输入数据等等，也会透过 X Server 来传达到 X Client 端，而由 X Client 来加以运算出应绘制的数据；

- X Client： 这组程序主要负责的是数据的运算。 X Client 在接受到 X Server 传来的数据后 (例如移动鼠标、点击 icon 等动作)，会经由本身的运算而得到鼠标应该要如何移动、点击的结果应该要出现什么样的数据、键盘输入的结果应该要如何呈现等等，然后将这些结果告知 X Server ，让他自行去绘制到屏幕上。

Tips: 鸟哥常常开玩笑的说， X server 就是画布，而 X client 就是手拿画笔的画家。你得要先有画布 (管理好所有可显示的硬件后) 之后画家的想法 (计算出来的绘图数据) 才能够绘制到画布上！



由于每一支 X client 都是独立存在的程序，因此在图形显示会发生一些迭图的问题 (想象一下每一个 X client 都是一个很自我的画家， 每个画家都不承认对方的存在，都自顾自的在画布上面作画，最后的结果会是如何？)。因此，后来就有一组特殊的 X client 在进行管理所有的

其他 X client 程序，这个总管的咚咚就是 Window Manager！

- Window Manager (WM)：是一组控制所有 X client 的管理程序，并同时提供例如任务栏、 背景桌面、虚拟桌面、窗口大小、窗口移动与重迭显示等任务。Window manager 主要由一些大型的计划案所开发而来，常见的有 GNOME, KDE, XFCE 等

既然 X Window System 是 Linux 上面的一组程序，那么它如何启动的呢？早期的用户在登入系统后，必须要自己先启动 X server 程序，然后再启动个别的 Window manager ，若有其他需求，再启动其他额外的 X client 就是了。这么麻烦！所以为了简化启动个人图形接口的步骤，后来还有所谓的 Display Manager (DM) 这玩意喔！

- Display Manager (DM)：提供使用者登入的画面以让用户可以藉由图形接口登入。 在使用者登入后，可透过 display manager 的功能去呼叫其他的 Window manager ，让用户在图形接口的登入过程变得更简单。 由于 DM 也是启动一个等待输入账号密码的图形数据，因此 DM 会主动去唤醒一个 X Server 然后在上头加载等待输入的画面就是了。

在目前新释出的 Linux distributions 中，通常启动图形接口让用户登入的方式中，都是先执行 Display Manager 程序， 该程序会主动加载一个 X Server 程序，然后再提供一个等待输入账号密码的接口程序，之后再根据用户的选择去启动所需要的 Window Manager 程序，最后就由用户直接操作 WM 来玩图形接口啰。

例题：在 CentOS 6.x 当中，若预设为 init 5 的情况下，那么最终启动图形接口的是哪一只程序？答：分析 /etc/init/* 当中的档案，会发现有个档案的内容是这样：

```
[root@www ~]# cat /etc/init/prefdm.conf
start on stopped rc RUNLEVEL=5
stop on starting rc RUNLEVEL=[!5]
console output
respawn
respawn limit 10 120
exec /etc/X11/prefdm -nodaemon
```

你可以分析 /etc/X11/prefdm 的内容，就能够发现其实该行启动的就是一个 X display manager 程序了喔！

例题：登入 init 5 的 CentOS 6.x 之前，先到 tty1 去查阅一下 X server 是由哪一支程序所唤醒的？答：我们可以透过 pstree 来观察程序间的相关性喔！同时注意，预设的 CentOS 6.x 的 X server 程序名称为 Xorg 的哩。

```
[root@www ~]# pstree -p
init(1)-+-NetworkManager(1086)
....(中间省略)....
        &#124;-gdm-binary(2642)---gdm-simple-slav(2661)-+-Xorg(2663)
        &#124;                                         &#124;-gdm-session-wor(2746)
....(后面省略)....
```

由上述的数据来看，gdm-binary 可以唤醒 Xorg 喔！同理，我们也会知道提供认证的图形画面应该是由 gdm-session 所提供的喔！

- X Window System 用在网络上的方式： XDMCP

当 X server, X client 都在同一部主机上面的时候，你可以很轻松的启动一个完整的 X Window System。 但是如果你想要透过这个机制在网络上面启动 X 呢？此时你得先在客户端启动一个 X server 将图形接口绘图所需要的硬件装置配置好， 并且启动一个 X server 常见的接收埠口(通常是 port 6000)，然后再由服务器端的 X client 取得绘图数据，再将数据绘制成图啰。 透过这个机制，你可以在任何一部启动 X server 登入服务器喔！而且不管你的操作系统是啥呢！意义就像下图， 如此一来，你就可以取得服务器所提供的图形接口环境啦！

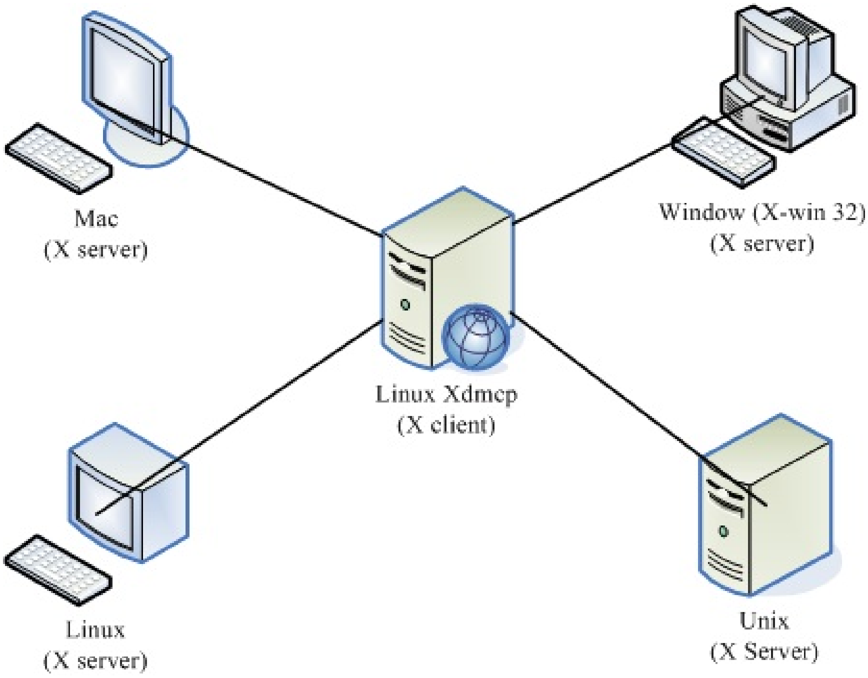


图 11.3-1、X server/client 的架构

但是如果你是使用最笨的方法在客户端自己启动 X server ，然后在告诉服务器将 X client 程序一个一个的加载回来， 那就太累人了吧！我们之前上面不是提到过可以用 display manager 来管理使用者的登入与启动 X 吗？那服务器能不能提供一个类似的服务， 那我们直接透过服务器的 display manager 就能够提供我们登入的认证与加载自己选择的 window manager 的话，这样就太棒了！ 能够达到吗？当然可以啊！那就是透过 Xdmcp (X display manager control protocol) (注3) 啦！

Xdmcp 启动后会在服务器的 udp 177 开始监听，然后当客户端的 X server 联机到服务器的 port 177 之后， 我们的 Xdmcp 就会在客户端的 X server 放上用户输入账密的图形接口程序啰！那你就能透过这个 Xdmcp 去加载服务器所提供的类似 Window Manager 的相关 X client 啰！那你就能够取得图形接口的远程联机服务器哩！赞吧！

那么什么时候会出现多使用者连入服务器取得 X 的情况呢？以鸟哥的例子来说，鸟哥实验室有一组 Linux 在进行数值模拟， 他输出的结果是 NetCDF 档案，我们必须使用 PAVE 这一套软件去处理这些数据。但是我们有两三个人同时都会使用到那个功能， 偏偏 Linux 主机是放

在机架柜里面的，要我们挤在那个小小的空间前面『站着』操作计算机，可真是讨人厌啊～这个时候，我们就会架设图形接口的远程登录服务器，让我们可以『多人同时以图形接口登入 Linux 主机』来操作我们自己的程序！很棒，不是吗！

11.3.2 设定 gdm 的 XDMCP 服务

既然是所谓的 Xdmcp 协议，那么是否意味着与 X display manager 有关呢？没错啦！ Xdmcp 协议是由 DM 程序所提供的。 我们的 CentOS 预设的 DM 为 GNOME 这个计划所提供的 gdm 哩！因此，你想要启动 Xdmcp 服务，那就得要针对 gdm 这个程序来设定啰。 这个 gdm 的设定数据都放置在 /etc/gdm/ 目录下，而我们所要修改的配置文件其实仅是一个 /etc/gdm/custom.conf (注4) 档案而已。

Tips: X11 提供的 display manager 为 xdm ，而著名的 KDE 与 GNOME 也都有自己的 display manager 管理程序，分别是 kdm 与 gdm 。你可以透过三者中任何一者的 display manager 的配置文件来启动 xdmcp 这个协定呢～



不过，因为我们安装的基准是『Basic server』，所以很多图形接口软件并没有被安装起来。因此，在实作 Xdmcp 之前，我们得先安装图形接口才行喔！使用 yum groupinstall 来安装吧！

```
# 先检查看看与 X 相关的软件群组有哪些？
[root@www ~]# yum grouplist
   Desktop
   Desktop Platform
   X Window System
# 这三个算是最重要的项目了！得要安装起来才行喔！gdm 是在 Destop 中！

[root@www ~]# yum groupinstall "Desktop" "Desktop Platform" \
&gt; "X Window System"
```

上面进行完毕后，现在才能开始搞定 custom.conf 啦！来试玩看看！

```
[root@www ~]# vim /etc/gdm/custom.conf
[security]           &lt;==在与资安方面有关的信息，大多指登录相关事宜
AllowRemoteRoot=yes  &lt;==xdmcp 预设不许 root 登入，得用这个项目才能以 root 登入
DisallowTCP=false    &lt;==这个项目在允许客户端使用 TCP 的方式联机到 xdmcp

[xdmcp]              &lt;==就是这个小节的重点之一啰！
Enable=true          &lt;==启动 xdmcp 的最重要项目啰！
# 上述特殊字体的部份就是你得要自己新增的内容啰！

[root@www ~]# init 5
# 上述这个指令会切换到 X 图形画面，如果确定要使用 gdm，runlevel 得调整到 5 才好
# 果真如此的话，那就得要调整 /etc/inittab 啰！

[root@www ~]# netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address  Foreign Address    State    PID/Program name
tcp        0      0 0.0.0.0:6000   0.0.0.0:*          LISTEN   4557/Xorg
tcp        0      0 :::6000        :::*               LISTEN   4557/Xorg
udp        0      0 0.0.0.0:177    0.0.0.0:*                   4536/gdm-binary
# 上述的 port 6000 是由 DisallowTCP=false 项目启动的，port 177 才是我们要的
```

上述的动作鸟哥是在 runlevel 3 底下启动的，如果你是在 runlevel 5 底下时，因此你也可以利用『 init 3 && init 5 』来重新启动图形接口。但如果你是在 runlevel 3 底下并且不希望变更成为 runlevel 5 呢？那又该如何启动 port 177 啊？如果是这样的话，那么你可以这样启动 xdmcp 啦：

```
[root@www ~]# init 3
[root@www ~]# runlevel
5 3 &lt;==左边的是前一个 runlevel，右边的是目前的，因此目前是 runlevel 3
[root@www ~]# gdm   &lt;==这样就启动 xdmcp 啰！
[root@www ~]# vim /etc/rc.d/rc.local
/usr/sbin/gdm
```

现在你知道如何在不同的 runlevel 启动 xdmcp 了吧？如果是 runlevel 5 ，因为在 /etc/inittab 就已经有自动启动 gdm 了， 所以你只要顺利启动 runlevel 5 即可。但如果你是在 runlevel 3 的话，因为这样 gdm 就不会被系统的启动流程启动， 那你只好自己在 /etc/rc.d/rc.local 里面指定启动他啰！这样了解呼？不过，既然你都要使用 xdmcp 了，所以建议您直接启动在 runlevel 5 即可！接下来，你得要开放客户端对你的 port 177 联机才行！ 请自行修改你的防火墙规则，开放 udp port 177 吧！鸟哥这里假设你使用鸟哥的防火墙脚本，那你这样作就好了：

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.rule
iptables -A INPUT -p UDP -i $EXTIF --dport 177 --sport 1024:65534 \
 -s 192.168.100.0/24 -j ACCEPT #xdmcp
# 注意喔！特点是使用 UDP 埠口以及加入来源端 IP 网域的控管！

[root@www ~]# /usr/local/virus/iptables/iptables.rule
[root@www ~]# iptables-save &#124; grep 177
-A INPUT -s 192.168.100.0/24 -i eth0 -p udp -m udp --sport 1024:65534 --dport 177 -j ACCE
# 确实有开放 port 177 ，而且是 udp 的埠口喔！要注意这两个项目。
```

11.3.3 用户系统为 Linux 的登入方式

由于 Linux 本身的窗口就是由 X server 提供来的，因此使用 Linux 登入远程的图形服务器是很简单的啦！ 但是因为启动 X 的方式不同而已数种启动方式，底下我们就讲讲两个常见的启动方式：

- 在不同的 X 环境下启动联机： 直接用 X

如果你的客户端已经在 runlevel 5 了，因此其实你已经有一个 X 窗口的环境，这个环境的显示终端机就称为『 :0 』。 在 CentOS 6.x 的环境中，如果原本就是 runlevel 5 的环境，那么这个图形接口的 :0 是在 tty1 终端机啦！如果是由 runlevel 3 启动图形接口，那就是在 tty7 喔！由于已经有一个 X 了，因此你必须要在另外的终端机启动另一个 X 才行！那个新的 X 就称为 :1 接口，其实通常就在 tty7 或 tty8 啦！但因为 X server 要接受 X client 必须要有授权才行， 所以你得先在窗口接口开放接受来自服务器的 X client 数据。

此外，虽然你在客户端是以主动的方式连接到服务器的 udp port 177 ，但是服务器的 X client 却会主动的连接到你客户端的 X server，因此，你必须要开放来自服务器端主动对你的 TCP port 6001 (因为是 :1 界面) 的防火墙联机才行喔！那就来实做看看：

```
# 1\. 放行 X client 传来的资料：在 X Window 的画面当中启用 shell 输入：
[root@clientlinux ~]# xhost + 192.168.100.254
192.168.100.254 being added to access control list
# 注意！你是客户端！且假设我刚刚那部 Linux 主机的 IP 为 192.168.100.254

# 2\. 开始放行防火墙，因为我们启动 port 6001 ，所以你在客户端这样作：
[root@clientlinux ~]# vim /usr/local/virus/iptables/iptables.allow
iptables -A INPUT -i $EXTIF -s 192.168.100.0/24 -p tcp --dport 6001 -j ACCEPT

[root@clientlinux ~]# /usr/local/virus/iptables/iptables.rule
[root@clientlinux ~]# iptables-save
-A INPUT -s 192.168.100.0/24 -p tcp -m tcp --dport 6001 -j ACCEPT
# 要能看到上面这一行才行呦！

# 3\. 在文字接口 (例如 tty1) 下输入如下的指令：
[root@clientlinux ~]# X -query 192.168.100.254 :1
# 进入 X Window 啰！
```

如果一切顺利的话，那么你在 clientlinux.centos.vbird 就会看到如下的画面(注意主机名)：

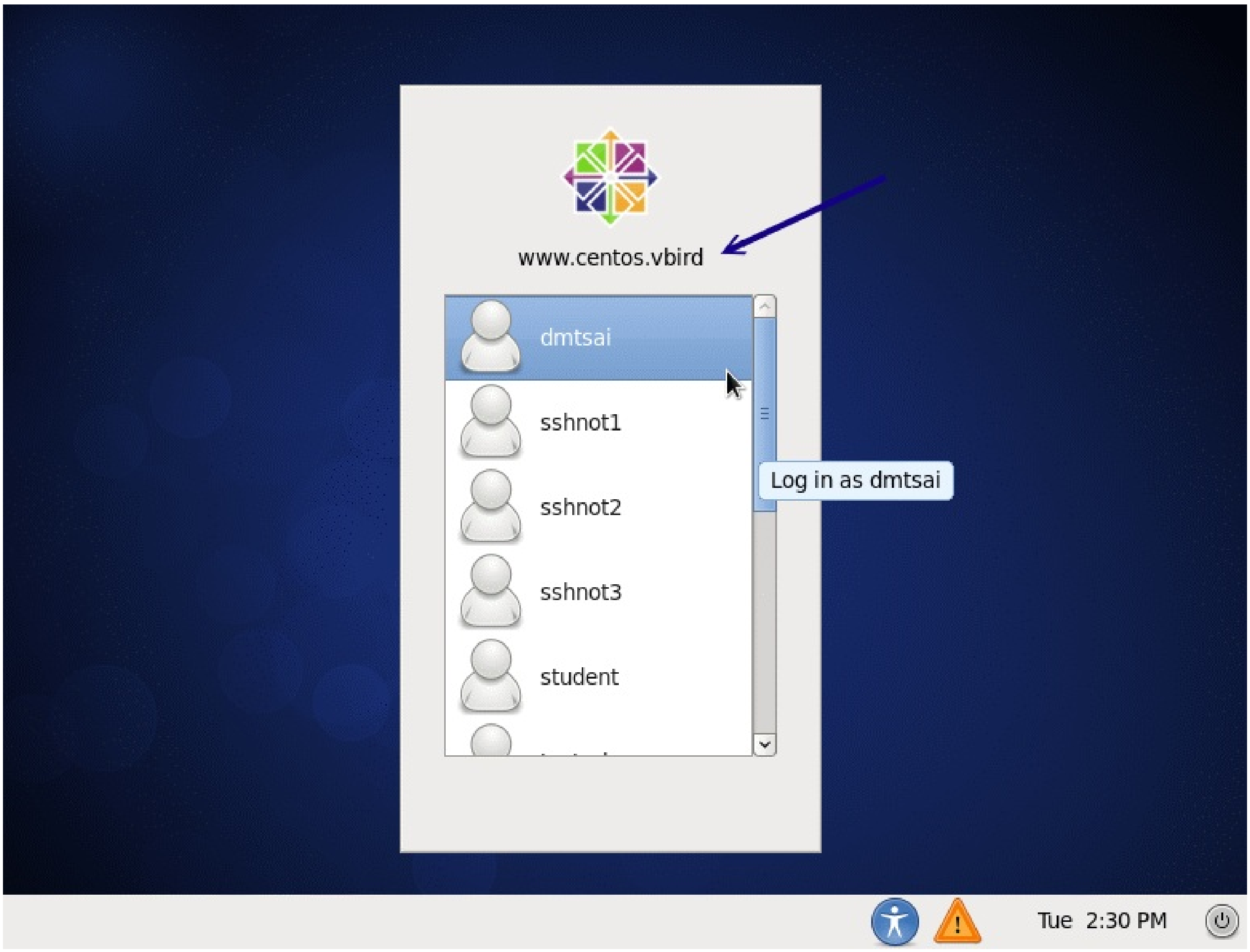


图 11.3-2、在客户端连上 Xdmcp 成功的画面

在上图中输入正确的账号与密码之后，你在 tty8 (:1) 就会有个窗口接口啰！那你如果想要回到本机的窗口接口， 就回到 tty7 (:0) 即可切换成功！(在 runlevel 5 时，:0 在 tty1 ，而 :1 在 tty7 喔！)那想要关闭 tty8 该如何是好？你不能够在 tty8 注销啦，因为注销后，系统会重新开一个等待登入的画面，你还是没办法关闭的。你得要回到刚刚启动 X 的 tty1 然后按下 [ctrl]-c 中断联机即可！

- 在同一个 X 底下启动另一个 X： 使用 Xnest

如果常常在 tty7, tty8 切换来去的话，偶而会忘记到底在哪个界面了，尤其是当你的桌面都一模一样时， 那就更难判断了。有没有办法直接在 tty7 启动另一个窗口来加载远程服务器的图形接口呢？可以的，那就透过 Xnest 吧！ 这指令需要在 X 的环境下使用喔！它的简单用法如下：

```
[root@www ~]# Xnest -query 主机名 -geometry 分辨率 :1
选项与参数：
-query     ：后面接 xdmcp 服务器的主机名或 IP 啰
-geometry  ：后面接画面的分辨率，例如 1024x768 或 800x600 等之类的分辨率

# 根据上述数据，使用 800x600 连上 192.168.100.254 那部主机：
[root@www ~]# yum install xorg-x11-server-Xnest
[root@www ~]# Xnest -query 192.168.100.254 -geometry 640x480 :1
```

如果一切顺利的话，那你就会在 tty7 的本机 X 环境下看到如下的画面 (底下的画面是已经登入的情况！)

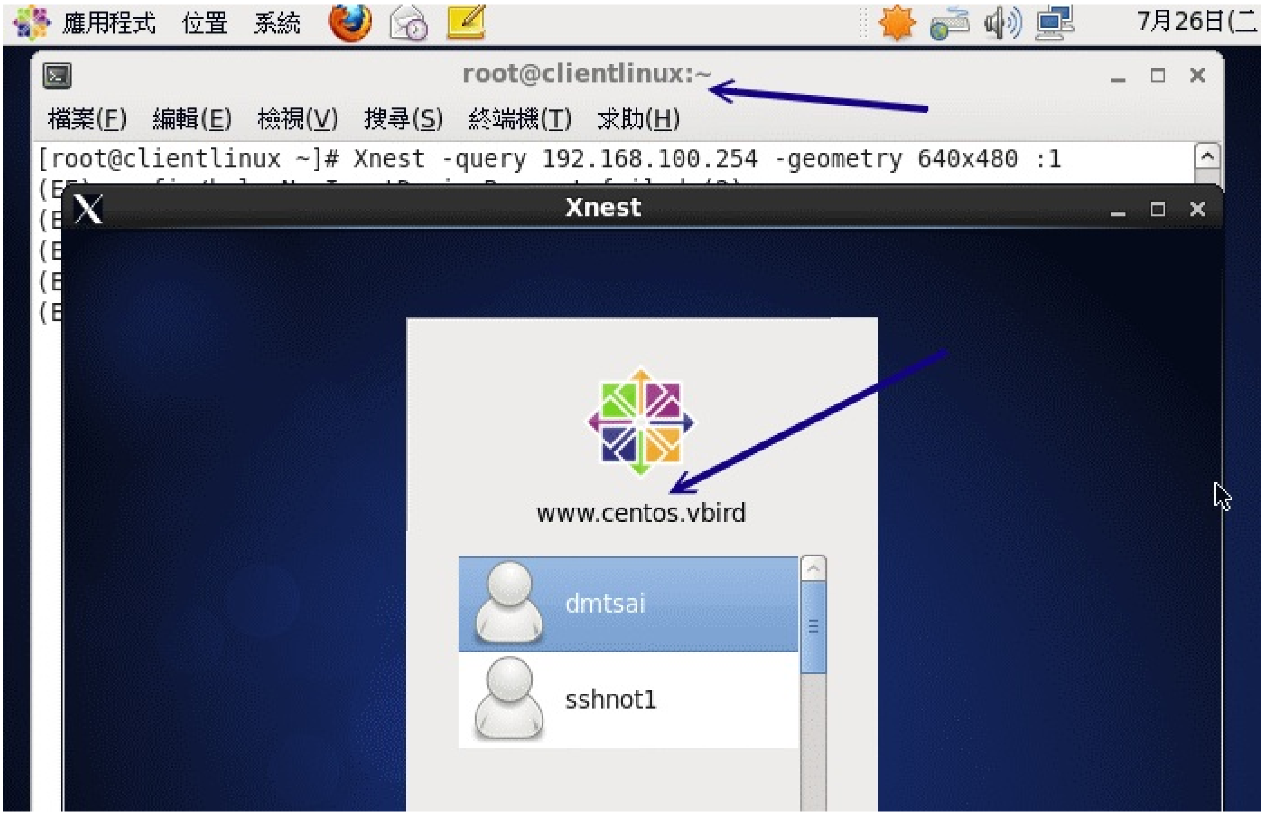


图 11.3-3、在客户端的 X 顺利连上 Xdmcp 的画面

一开始的图示会与图 11.3-2 一样，就是出现输入账密的画面，如果输入正确的帐密后， 就会出现上述的图示了。仔细看一下画面当中的终端机标头，你就会发现确实是两部主机的桌面呢！这样有没有更棒棒？ ^_^！ 要关闭这个 X 就简单多了！直接按下关闭，或者是中断那个 Xnest 的程序即可。

11.3.4 用户系统为 Windows 的登入方式： Xming

由于 Windows 本身并没有提供预设的 X server ，因此我们得要自行安装 X server 在 Windows 上面才行。 目前常见的 X server 有底下这几个：

- X-Win32 (http://www.xwin32.tw/)
- Exceed (http://www.hummingbird.com/products/nc/exceed/index.html?cks=y)
- Xming (http://sourceforge.net/projects/xming/)

其中 X-Win32 与 Exceed 都属于商业软件，而 Xming 则属于轻量级的自由软件，说是轻量级并非说它不好， 而是因为 Xming 的档案真的很小，而该有的功能都有了，所以算是很不赖的一个软件喔！因此底下鸟哥是以 Xming (注5) 作范例来介绍的。

1. 安装：你可以使用预设的方法，一直下一步的安装下去，就能够顺利的安装好 Xming 这套 X server 的软件啰。

2. 启动：请在『开始』--> 『程序集』--> 『Xming』--> 『XLaunch』开启设定联机到 xdmcp 的方式。底下我们会使用区网内的广播 (broadcast) 来找到 xdmcp 服务器的方式。启动 XLaunch 之后会出现如下的图示：

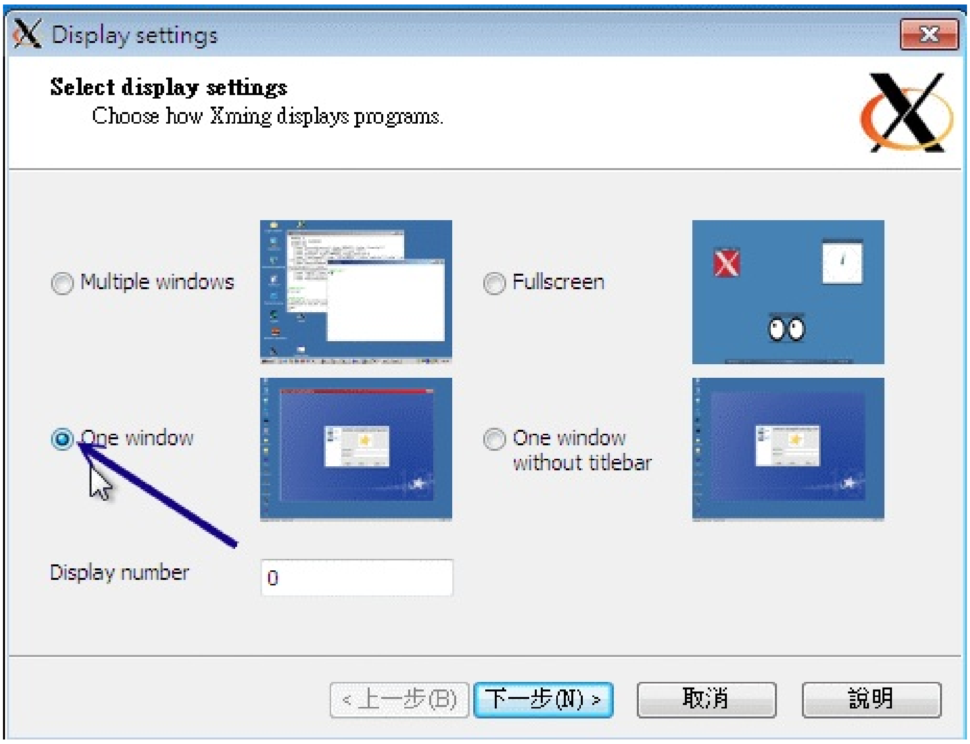


图 11.3-4、Xming 的 Xdmcp 连接方式示意图

记得上面的图示要选择 One window 或 Fullscreen 或 One window without titlebar 才能够使用 XDMCP 喔！选择完毕后按『下一步』 就会出现如下的画面：

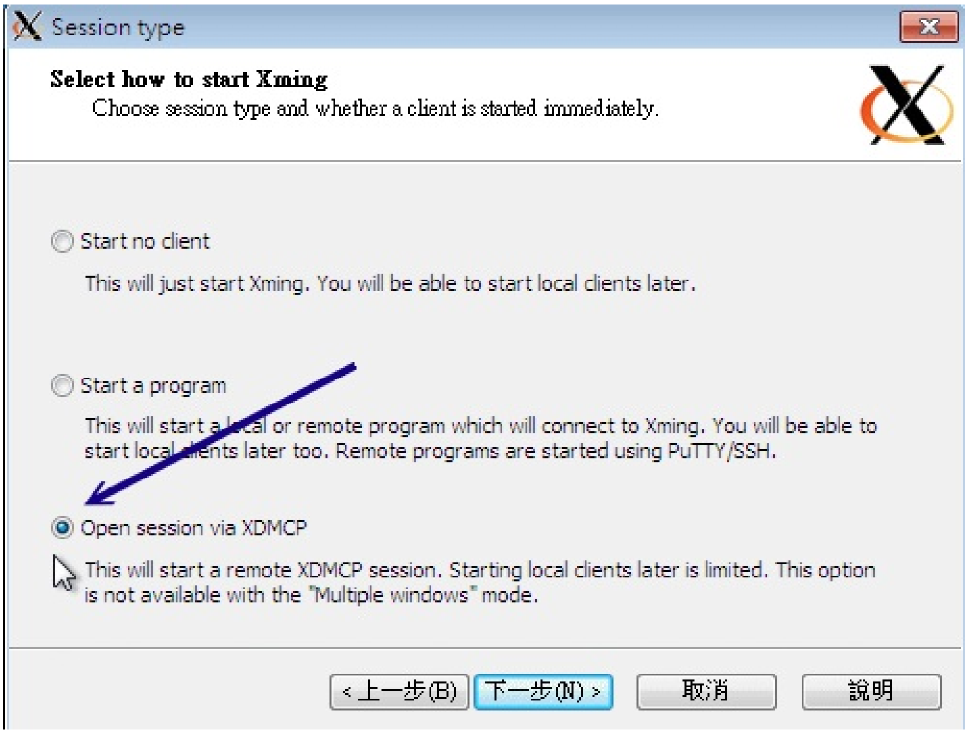


图 11.3-5、Xming 的 Xdmcp 连接方式示意图

上述的图示当中共有三种传递 X client 的方法，在这个小节当中我们要连到 xdmcp ，所以你得要选择第三个喔！之后再下一步会出现下图：

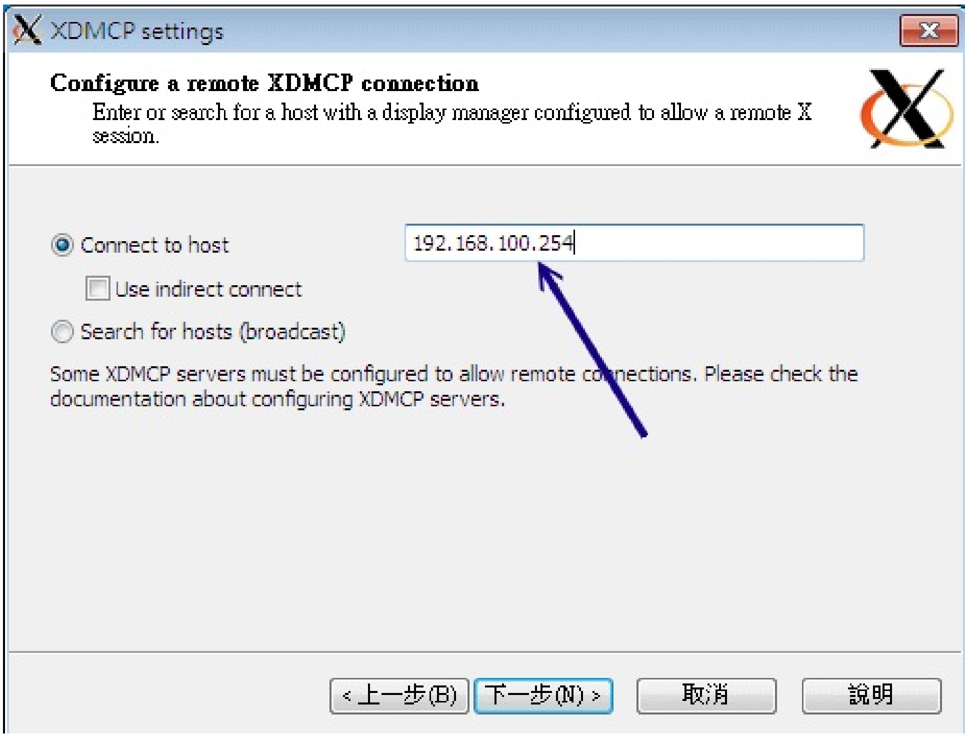


图 11.3-6、Xming 的 Xdmcp 连接方式示意图

这里当然就是连接到你想要连上去的 xdmcp 服务器啰！将他的 IP 填上去吧！之后再下一步去：

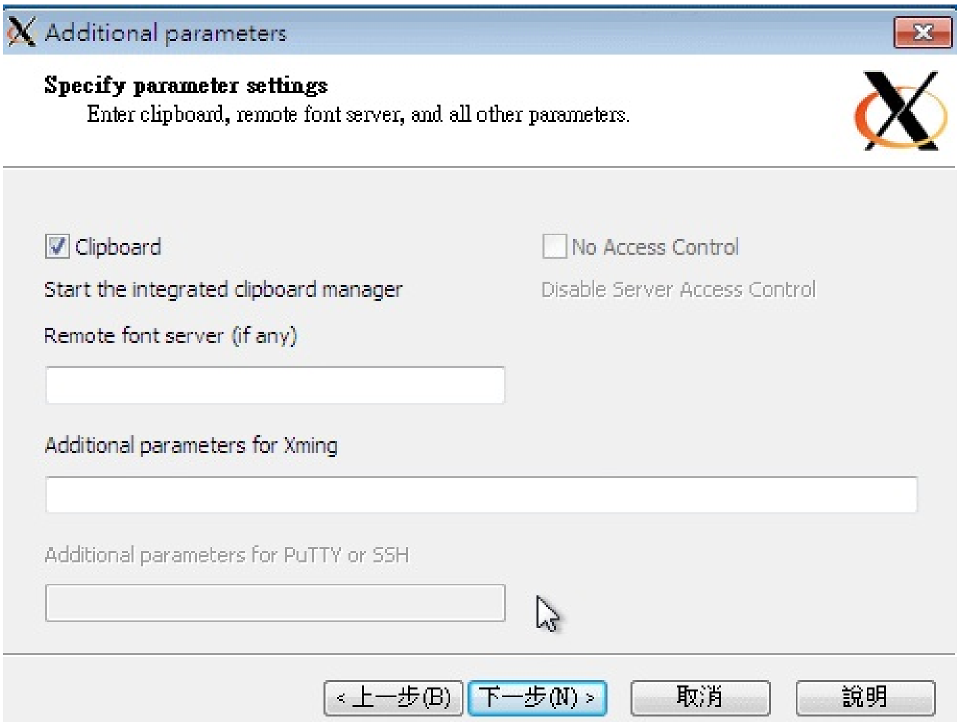


图 11.3-7、Xming 的 Xdmcp 连接方式示意图

上图的项目与数据的互相复制贴上有关，保留默认值即可。按下下一步吧！

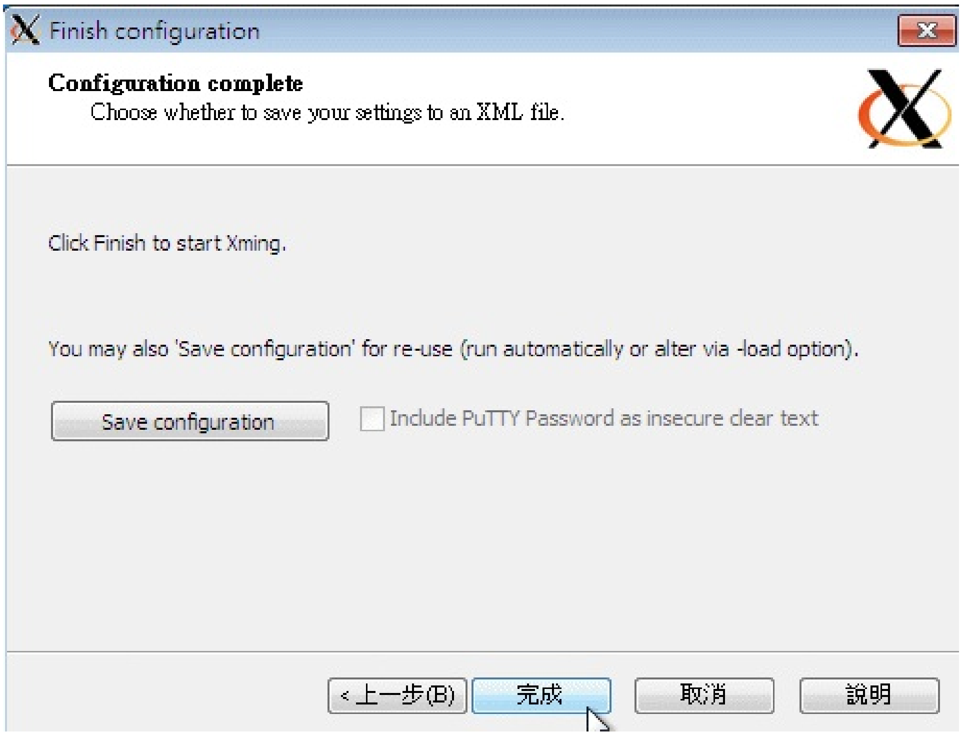


图 11.3-8、Xming 的 Xdmcp 连接方式示意图

出现上图就是设定完毕了，给它按下『完成』之后，你就会发现如同图 11.3-2 的画面出现，你就能够开始在 Windows 底下连上图形接口的 Linux Server 啰！很轻松吧！

3. 重点在 Server 与 Client 的防火墙上

其实从上面的设定当中你会发现， XDMCP 不论是在 Server 还是 Client 的设定上面都很简单！但是有时候你就是会发现， 明明所有的动作都做完了，但是就是没有办法连上 Xdmcp 服务器！最容易发生错误的其实就是防火墙啦！因为虽然我们客户端启动 X server 后，会主动联机到服务器端的 Xdmcp (port 177)，但是，接下来却是服务器主动联机到我们客户端的 X server (可能是 port 6000~6010)。 因此，如果你只是设定了服务器的防火墙而已，那么很可能出现问题的应该就是客户端的防火墙忘记打开提供服务器主动联机的规则啰！ 这点是必须要跟大家说明的喔！

11.4 华丽的图形接口： VNC 服务器

就如同刚刚上头讲到的，使用 xdmcp 可能会启动多个不同的埠口，导致防火墙设定上面比较困扰些。那有没有简单一点的图形接口连接方式？ 其实还有很多啦，在这里我们先来讲一个比较简单的，那就是 VNC (Virtual Network Computing) 这玩意儿啦！(注6)

11.4.1 预设的 VNC 服务器：使用 twm window manager

VNC server 会在服务器端启动一个监听用户要求的端口，一般端口号码在 5901 ~ 5910 之间。当客户端启动 X server 联机到 5901 之后， VNC server 再将一堆预先设定好的 X client 透过这个联机传递到客户端上，最终就能够在客户端显示服务器的图形接口了。

不过需要注意的是，预设的 VNC server 都是独立提供给『单一』一个客户端来联机的，因此当你要使用 VNC 时， 再联机到服务器去启动 VNC server 即可。所以，一般来说， VNC server 都是使用手动启动的，然后使用完毕后， 再将 VNC server 关闭即可。整个作法其实很简单喔！你可以这样作：

```
[root@www ~]# vncserver [:号码] [-geometry 分辨率] [options]
[root@www ~]# vncserver [-kill :号码]
选项与参数：
:号码       ：就是将 VNC server 开在哪个埠口，如果是 :1 则代表 VNC 5901 埠口
-geometry  ：就是分辨率，例如 1024x768 或 800x600 之类的
options    ：其他 X 相关的选项，例如 -query localhost 之类的
-kill      ：将已经启动的 VNC 埠口删除！依据身份控制喔。

[root@www ~]# yum install tigervnc-server
# 这个是必须要的服务器软件，注意软件的名称喔！与之前的版本不同！

# 将 VNC server 启动在 5903 埠口
[root@www ~]# vncserver :3

You will require a password to access your desktops.

Password:  &lt;==输入 VNC 的联机密码，这是建立 VNC 时所需要的
Verify:    &lt;==再输入一次相同的密码
xauth:  creating new authority file /root/.Xauthority

New 'www.centos.vbird:3 (root)' desktop is www.centos.vbird:3

Creating default startup script /root/.vnc/xstartup
Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/www.centos.vbird:3.log

[root@www ~]# netstat -tulnp &#124; grep X
tcp        0      0 0.0.0.0:5903   0.0.0.0:*      LISTEN      4361/Xvnc
tcp        0      0 0.0.0.0:6000   0.0.0.0:*      LISTEN      1755/Xorg
tcp        0      0 0.0.0.0:6003   0.0.0.0:*      LISTEN      4361/Xvnc
tcp        0      0 :::6000        :::*           LISTEN      1755/Xorg
tcp        0      0 :::6003        :::*           LISTEN      4361/Xvnc
# 已经启动所需要的埠口啰！
```

在上述的指令操作中，你要知道的几个项目是：

1. 密码至少需要六个字符
2. 依据使用 vncserver 的身份，将刚刚建立的密码放置于该账号家目录下。例如上述的身份是使用 root 身份，因此密码文件会放在 /root/.vnc/passwd 这个档案中但是若该档案已经存在，则不会出现建立密码的画面。
3. 当客户端联机成功后，服务器将会传送 /root/.vnc/startx 内的 X client 给客户端喔！

那如果你想要修改 VNC 密码呢？很简单，那就使用 vncpasswd 吧！

```
[root@www ~]# ls -l /root/.vnc/passwd
-rw-------. 1 root root 8 Jul 26 15:08 /root/.vnc/passwd
[root@www ~]# vncpasswd
Password:  &lt;==就是这里开始输入新的密码啊！
Verify:
[root@www ~]# ls -l /root/.vnc/passwd
-rw-------. 1 root root 8 Jul 26 15:15 /root/.vnc/passwd
# 看吧！时间有更新喔！这个档案的内容更动过啰！
```

接下来开始放行 5903 这个埠口的联机防火墙规则吧！因为预计可能会开放 11 个 VNC 的埠口，所以干脆一口气开放 11 个埠口吧！

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.allow
iptables -A INPUT -i $EXTIF -s 192.168.100.0/24 -p tcp --dport 5900:5910 -j ACCEPT

[root@www ~]# /usr/local/virus/iptables/iptables.rule
[root@www ~]# iptables-save
-A INPUT -s 192.168.100.0/24 -i eth0 -p tcp -m tcp --dport 5900:5910 -j ACCEPT
# 要看得到上面这行才 OK 喔！
```

11.4.2 VNC 的客户端联机软件

与 xdmcp 很类似啦， VNC 客户端在 Linux 系统上面有默认的软件，但是在 Windows 系统上面则必须要额外安装其他软件。 我们先来谈谈 Linux 的 VNC 用户软件吧！

- Linux 客户端程序： vncviewer

用在 Linux 客户端的 VNC 程序，那就是 vncviewer。只是，这个软件默认没有安装，所以你得要使用 yum 安装完毕后再来联机吧！不过一样要注意，服务器端的防火墙一样要设定妥当喔！然后开始在客户端的图形接口上执行底下数据：

```
[root@clientlinux ~]# yum install tigervnc
[root@clientlinux ~]# vncviewer 192.168.10.254:3
# 这个指令请一定一定要在图形接口上面执行才行喔！很重要！别忘了！
```

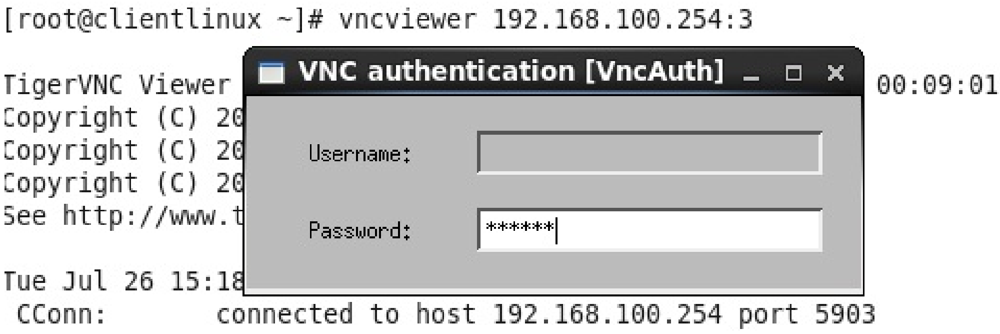


图 11.4-1、在 Linux 客户端执行 vncviewer 程序示意

在上图当中输入刚刚的 root 的 VNC 联机密码，请注意喔，是 VNC 的联机密码，而不是 root 的登入密码！ 这两者是差很多的！也由于启动 VNC 的身份是 root ，因此这里才使用 root 的 VNC 联机密码。 所以，很多时刻，我们都是建议使用一般身份来启动 VNC server 的啦！当你输入正确的 VNC 联机密码后， 会出现如下的图示啰：

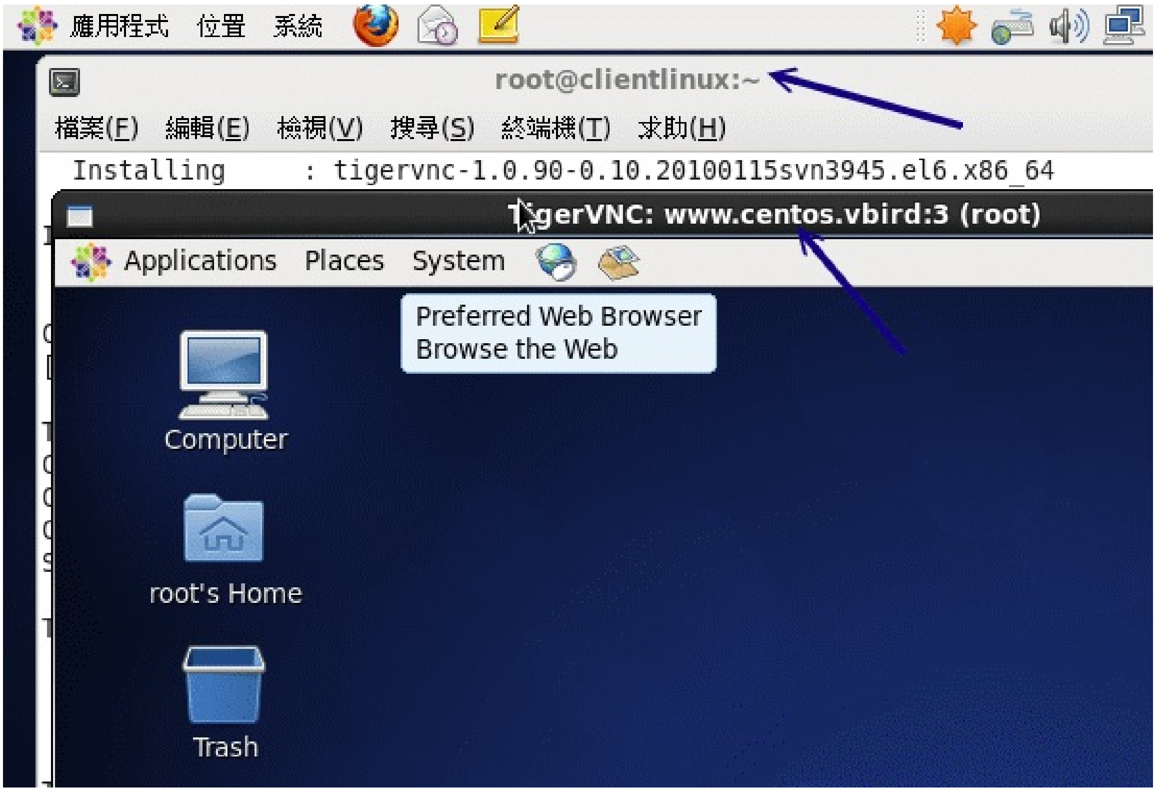


图 11.4-2、在 Linux 客户端执行 vncviewer 程序示意

与以前的 VNC server 较大的差异，在 CentOS 6.x 当中，tigervnc-server 这套软件会主动的依据服务器端的图形接口登入方式给予正确的图形显示接口，而不是以前那样给予一个丑丑的 twm 而已！ 这样我们就可以减少还得要修改一些有的没有的配置文件了！真是棒！ 联机成功后，请在客户端关闭这个 vncviewer 的联机，因为接下来我们要准备由 Windows 联机到服务器的 port 5903 啰！

- Windows 客户端程序： realvnc

Windows 底下可用的 vnc client 软件不少，但是鸟哥比较熟悉的是 realvnc 这家公司出品的 GNU 的自由软件！ 你可以在底下的连结下载到最简单的版本，是不用钱的自由软件版本喔！(鸟哥仅下载不用安装的 viewer 版本而已！)

- http://www.realvnc.com/download.html

直接执行 vnc-viewer 软件，然后就会看到如下的画面：

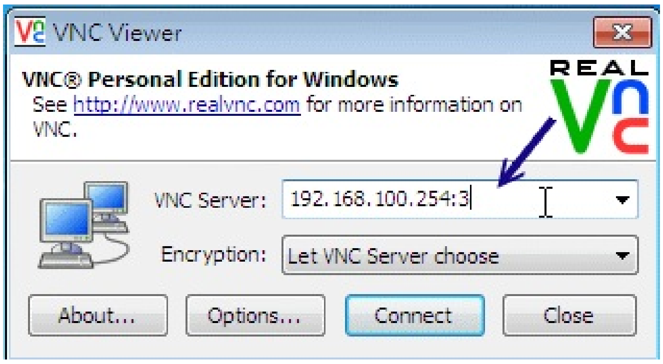


图 11.4-3、Windows Real VNC 客户端联机示意图

如上图所示，你在 server 字段填上 IP:port 的数据即可，然后按下『OK』吧！

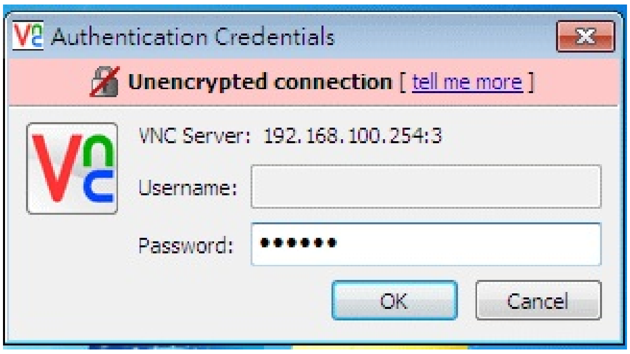


图 11.4-4、Windows Real VNC 客户端联机示意图

由于 VNC server 需要的仅是联机的 VNC 密码而已，因此上图中的 Username 可以不用填，老实说，这个程序它也不会让你填～ 呵呵！填完按下『OK』即可！接下来就会出现正确的画面啰！

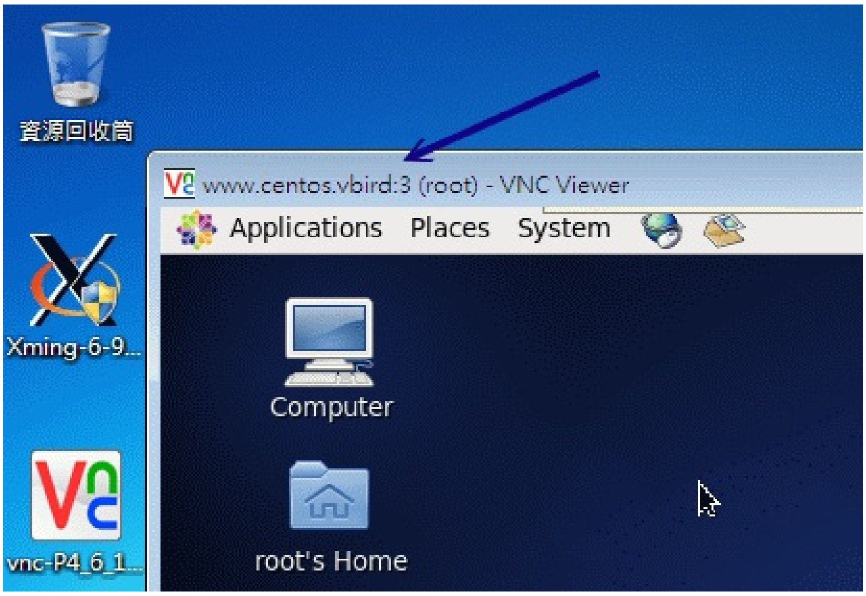


图 11.4-5、Windows Real VNC 客户端联机示意图

11.4.3 VNC 搭配本机的 Xdmcp 画面

如果因为某些特殊因素，你得要使用 VNC 来搭配 xdmcp 的输出时，那就直接在服务器透过底下的指令来处理即可！ 要注意喔，你必须要已经启动了 xdmcp 了喔！而且，我们底下使用 student 的身份来启动这个 VNC 吧！

```
# 1\. 要确定 xdmcp 已经启动了才可以：
[root@www ~]# netstat -tlunp &#124; grep 177
udp        0      0 0.0.0.0:177    0.0.0.0:*       1734/gdm-binary
# OK 的！确实有启动的啦！如果没有看到 177 的话，回到 [11.3](#xdmcp) 去处理处理

# 2\. 切换成 student，并且启动 VNC server 在 :5
[root@www ~]# su - student
[student@www ~]$ vncserver :5 -query localhost
You will require a password to access your desktops.

Password:
Verify:
xauth:  creating new authority file /home/student/.Xauthority

New 'www.centos.vbird:5 (student)' desktop is www.centos.vbird:5

Creating default startup script /home/student/.vnc/xstartup
Starting applications specified in /home/student/.vnc/xstartup
Log file is /home/student/.vnc/www.centos.vbird:5.log

# 3\. 取消 xstartup 的启动内容
[student@www ~]$ vim /home/student/.vnc/xstartup
....(前面省略)....
#xterm -geometry 80x24+10+10 -ls -title "$VNCDESKTOP Desktop" &
#twm &
# 将这个档案的内容，全部都加上 # 批注掉

# 4\. 重新启动 vncserver 喔！
[student@www ~]$ vncserver -kill :5
[student@www ~]$ vncserver :5 -query localhost
```

接下来请使用 root 的身份加入 5905 的端口防火墙规则，然后自行使用 Linux 的 vncviewer 或 Windows 的 RealVNC 来联机，你就会发现如下的画面：

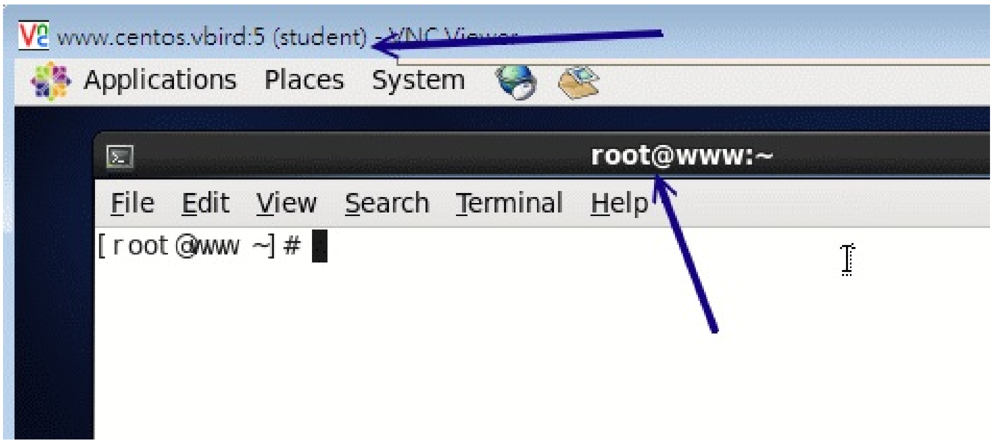


图 11.4-6、透过 VNC 通道取得 xdmcp 画面

我们这只 VNC 的联机程序是 student 身份，但是我们却可以透过 xdmcp 的登入功能来登入 root 身份喔！ 因为在服务器上面的 Xvnc 程序是 student 拥有，这样会比较好啦！了解呼？

11.4.4 开机就启动 VNC server 的方法

请注意，你不要将 vncserver 的指令写入在 /etc/rc.d/rc.local 中，否则可能会产生 localhost 无法登入的问题。 那该如何让你的 VNC server 在一开机就启动而不须要登入执行指令呢？可以的，但是你得要修改一下配置文件。 我们底下使用 student 的身份启动 VNC server，而启动的方式为使用 xdmcp 登入画面，启动的埠口就定在 5901 好了。 那你应该这样作：

```
[root@www ~]# vim /etc/sysconfig/vncservers
VNCSERVERS="1:student"
VNCSERVERARGS[1]="-query localhost"
# 上述两行的 1 指的就是那个埠口 5901 喔！要注意！

[root@www ~]# /etc/init.d/vncserver restart
[root@www ~]# chkconfig vncserver on
```

有够好简单吧！这样每次开机就搞定你的 VNC server 啰！

11.4.5 同步的 VNC ：可以透过图示同步教学

另外，有些朋友一定会觉得奇怪，那就是，为甚么我的 VNC 服务器的 server / client 端画面并不是同步的呢? 这是因为 Linux 本身提供多个 VNC server ，她们是各自独立的，所以当然就不会与 tty7 的画面同步了。 但是如果你想要与 Linux 的 tty7 同步的话，可以利用 VNC 释出的给 X Server 使用的模块来加以设定即可。

那使用这个模块有甚么好处啊？就是可以让两个图形接口在 server/client 都是一样的， 所以，如果你想要教你的朋友你是如何设定的，那就可以透过这个机制来处理，你的朋友在远程就能够知道你一步一步进行的过程！ 这样很不赖吧！详细的作法可以参考底下的连结：

- http://phorum.study-area.org/viewtopic.php?t=25713

我们也来实做一下吧 (在 CentOS 6.x 当中并没有 xorg.conf 这个配置文件喔！所以，如果你要使用这些数据的话， 恐怕得要自行使用 X -configure 去建置 xorg.conf 后，再挪到 /etc/X11/ 去，然后才改的到设定！)：

```
[root@www ~]# yum install tigervnc-server-module
[root@www ~]# vim /etc/X11/xorg.conf
Section "Screen"
        Identifier "Screen0"
        Device     "Videocard0"
        DefaultDepth     24
        # VBird
        Option "passwordFile" "/home/student/.vnc/passwd"
        SubSection "Display"
                Viewport   0 0
                Depth     24
        EndSubSection
EndSection

# VBird
Section "Module"
    Load    "vnc"
EndSection
# 假设你的 vnc 密码档案放置在 /home/student/.vnc/passwd 里头，
# 这个时候就得要将密码文件内容写到 Screen 这个 section 当中了

[root@www ~]# init 3 ; init 5
[root@www ~]# netstat -tlunp &#124; grep X
tcp        0      0 0.0.0.0:5900    0.0.0.0:*      LISTEN      7445/Xorg
tcp        0      0 0.0.0.0:6000    0.0.0.0:*      LISTEN      7445/Xorg
tcp        0      0 :::6000         :::*           LISTEN      7445/Xorg
# 注意看喔！这几个 port 启动的 PID 都一样喔！所以会启动一个 port 5900 啰！
```

之后你可以使用『 vncviewer 192.168.100.254 』来联机即可，不需要加上 :0 之类的埠口。然后你可以看一下客户端与服务器端的图形接口，你会发现到两者移动鼠标时，两者的画面会同步运作喔！ 非常有趣呢！只不过这个动作还是只允许一条 VNC 联机，不能让所有客户端都连到 port 5900 ，这真是太可惜了！

11.5 仿真的远程桌面系统： XRDP 服务器

使用上面的图形接口的联机服务器都有一个问题，除了联机机制的不同之外，上头的 Xdmcp 与 VNC 原则上，资料都没有加密。 因此上面的动作大多仅适合局域网络内运作，不要连上 Internet 比较好。那如果你真的想要透过加密的方式运作 VNC， 那可能得要透过下一小节的介绍才能够有好的处理结果。那么我们知道 Windows 的远程桌面 (Remote Desktop Procotol, RDP, 注7) 其实是具有联机加密功能的，所以，能不能在 Linux 上面装一个 RDP Server 呢？是可以的，那就是 XRDP 服务器 (注8)。

很可惜的是，我们的 CentOS 6.x 预设并没有提供 XRDP 的服务器，如果你有兴趣的话，可以自行编译 xrdp 软件， 但鸟哥有找到 Fedora 基金会提供的 RHEL 额外软件计划 (注9)，你可以到底下的连结去找到你对应的版本：

- http://download.fedora.redhat.com/pub/epel/

鸟哥还是觉得 yum 是好东西，因此鸟哥找到的 CentOS 6.x x86_64 版本的网址后，将它设定在 yum 配置文件内，就可以使用 yum 安装了：

```
[root@www ~]# vim /etc/yum.repos.d/fedora_epel.repo
[epel]
name=CentOS-$releasever - Epel
baseurl=http://download.fedora.redhat.com/pub/epel/6/x86_64/
gpgcheck=0
enabled=1

[root@www ~]# yum clean all
[root@www ~]# yum install xrdp
```

这样就安装好了 xrdp 软件了，接着下来就得要开始来设定它啰！老实说，在一般的主机上面安装好这个 xrdp 之后，你根本不需要调整任何配置文件，保留好配置文件就好了，然后启动它，并且设定开机后启动，未来只要用远程联机连到这部主机， 系统就会启动 5910~5920 以上的 VNC 埠口，然后你就能够透过 RDP 的协议取得 VNC 的画面，最后就能够登入系统啰！

```
[root@www ~]# /etc/init.d/xrdp start
[root@www ~]# chkconfig xrdp on
[root@www ~]# netstat  &#124; grep xrdp
tcp        0      0 127.0.0.1:3350   0.0.0.0:*      LISTEN     6615/xrdp-sesman
tcp        0      0 0.0.0.0:3389     0.0.0.0:*      LISTEN     6611/xrdp
# 远程桌面的埠口是 3389 ，但是 xrdp 会再连到本机的 3350 去唤醒一个 VNC 的联机。
# 但是尚未联机之前，并不会起动任何的 VNC 埠口就是了。
```

如果你是使用 Windows 系统，那么透过『开始』--> 『程序集』--> 『附属应用程序』--> 『远程桌面联机』， 在出现的画面中输入这部 xrdp 服务器的 IP 之后，如果顺利连上就会出现如下的画面：

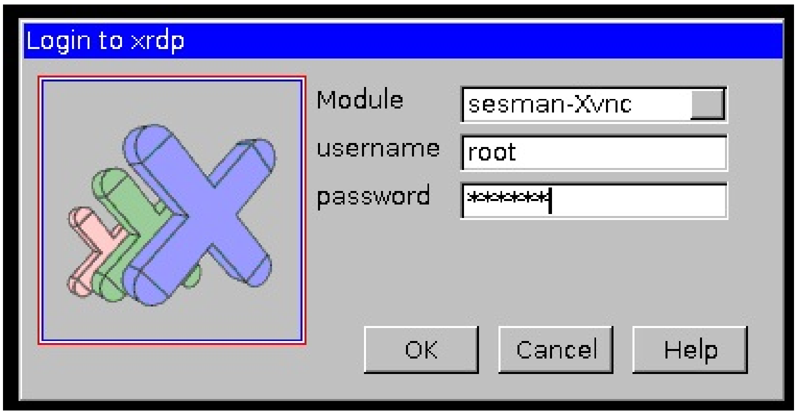


图 11.5-1、连上服务器的 XRDP 服务后，会出现的联机信息

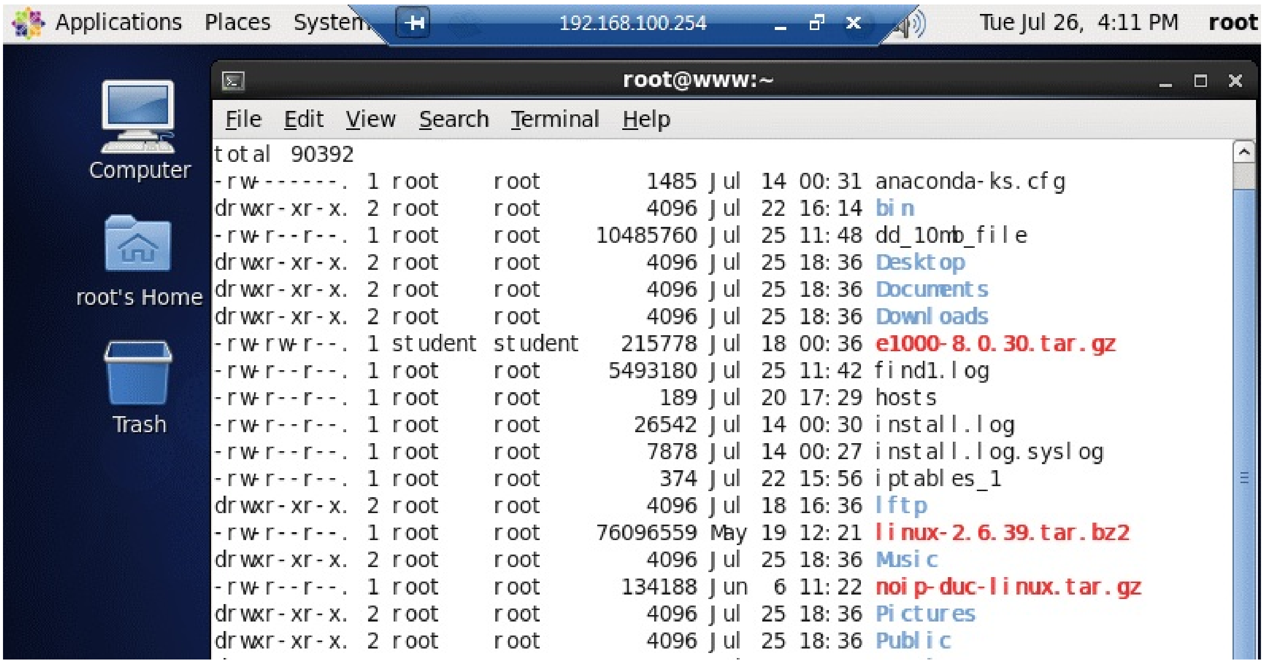


图 11.5-2、连上服务器的 XRDP 服务后，会出现的联机信息

输入正确的账号密码，嘿嘿！搞定！画面就出现啰！如果你还想要更进一步的了解 xrdp 的配置文件，那么请到 /etc/xrdp/ 目录底下瞧瞧，然后再透过 man 去看看相关的配置文件信息，就能够理解设定值啰！鸟哥测试过，不用修改任何设定， 使用远程桌面就已经很顺畅啰！^_^

不过你要注意的是，因为 xrdp 最终会自动启用 VNC ，因此你还是必须要安装 tigervnc-server 才行！ 否则 xrdp 应该还是无法运作的呦！

11.6 SSH 服务器的进阶应用

事实上 ssh 眞的很好用！你甚至不需要启动甚么 xdmcp, vnc, xrdp 等等服务，使用 ssh 的加密通道就能够在客户端启动图形接口！ 此外，我们知道很多服务都是没有加密的，那么能不能将这些服务透过 ssh 通道来加密呢？嘿嘿！当然是可以！ 在这个章节当中，我们就来谈谈一些 ssh 的进阶应用吧！

11.6.1 启动 ssh 在非正规埠口 (非 port 22)

从前面的章节里面我们就曾经提过， sshd 这个服务其实并不是很安全，所以很多 ISP 在入口处就已经将 port 22 关闭了！为什么要这么作呢？这是因为很多网站管理员并没有定期的进行软件 update ，而且为了方便，又很开心的将 port 22 对全世界开放。由于很多 cracker 会使用扫描程序乱扫整个 Internet 的埠口漏洞，这个 port 22 就是一个很常被扫描的端口啦！为了杜绝这个问题，所以 ISP 先帮你把关，先将 port 22 关闭！这也是为了整个区网好！

只是，像鸟哥这种没有 ssh 就快要活不下去的人，关闭了 port 22 那鸟哥的头都痛了！没有办法工作啊！ 那怎办？没关系，其实我们可以将 ssh 开放在非正规的埠口。如此一来， cracker 不会扫描到该端口，而你的 ISP 又没有对该埠口进行限制，那你就能够使用 ssh 啰！很棒吧！那就来试看看。我们底下将 ssh 开放在 port 22 及 port 23 试看看 (请注意， port 23 不能够有被使用喔！)。

- 设定 ssh 在 port 22 及 23 两个埠口的设定方式

```
[root@www ~]# vim /etc/ssh/sshd_config
Port 22
Port 23    &lt;==注意喔！要有两个 Port 的设定才行！

[root@www ~]# /etc/init.d/sshd restart
```

但是这一版的 CentOS 却将 SSH 规范 port 仅能启动于 22 而已，所以此时会出现一个 SELinux 的错误！那怎办？没关系， 根据 setroubleshoot 的提示，我们必须要自行定义一个 SELinux 的规则放行模块才行！有没有很难呢？其实还算简单！ 整体流程是这样的：

```
# 1\. 于 /var/log/audit/audit.log 找出与 ssh 有关的 AVC 信息，并转为本地模块
[root@www ~]# cat /var/log/audit/audit.log &#124; grep AVC &#124; grep ssh &#124; \
&gt;  audit2allow -m sshlocal &gt; sshlocal.te  &lt;==扩展名要是 .te 才行
[root@www ~]# grep sshd_t /var/log/audit/audit.log &#124; \
&gt;  audit2allow -M sshlocal  &lt;==sshlocal 就是刚刚建立的 .te 檔名
******************** IMPORTANT ***********************
To make this policy package active, execute:
semodule -i sshlocal.pp   &lt;==这个指令会编译出这个重要的 .pp 模块！

# 2\. 将这个模块加载系统的 SELinux 管理当中！
[root@www ~]# semodule -i sshlocal.pp

# 3\. 再重新启动 sshd 并且观察埠口吧！
[root@www ~]# /etc/init.d/sshd restart
[root@www ~]# netstat -tlunp &#124; grep ssh
tcp        0      0 0.0.0.0:22    0.0.0.0:*    LISTEN      7322/sshd
tcp        0      0 0.0.0.0:23    0.0.0.0:*    LISTEN      7322/sshd
tcp        0      0 :::22         :::*         LISTEN      7322/sshd
tcp        0      0 :::23         :::*         LISTEN      7322/sshd
```

有没有很简单！这样你就能够使用 port 22 或 port 23 联机到你的 sshd 服务喔！

- 非正规埠口的联机方式

由于预设的 ssh, scp, sftp 都是连接到 port 22 的，那么如何使用这些指令联机到 port 23 呢？我们使用 ssh 当练习好了：

```
[root@www ~]# ssh -p 23 root@localhost
root@localhost's password:
Last login: Tue Jul 26 14:07:41 2011 from 192.168.1.101
[root@www ~]# netstat -tnp &#124; grep 23
tcp  0  0 ::1:23            ::1:56645            ESTABLISHED 7327/2
tcp  0  0 ::1:56645         ::1:23               ESTABLISHED 7326/ssh
# 因为网络是双向的，因此自己连自己 (localhost)，就会抓到两只联机！
```

这样，你就能够避过一些 ISP 或者是 cracker 的扫描了！注意一下，不要将 port 开放在某些既知的埠口上， 例如你开放在 port 80 的话，那你就没有办法启动正常的 WWW 服务啦！注意注意！

11.6.2 以 rsync 进行同步镜像备份

我们曾在基础篇第三版第二十五章里头谈到 Linux 的备份策略， 该篇曾介绍常用的备份指令，包括 tar, dd, cp 等等，不过当时并未介绍网络，所以有个很棒的网络工具没有介绍， 那就是这个地方要谈到的 rsync 啦！这个 rsync 可以作为一个相当棒的异地备援系统的备份指令喔！ 因为 rsync 可以达到类似『镜相 (mirror) 』的功能呢！

rsync 最早是想要取代 rcp 这个指令的，因为 rsync 不但传输的速度快，而且他在传输时， 可以比对本地端与远程主机欲复制的档案内容，而仅复制两端有差异的档案而已，所以传输的时间就相对的降低很多！ 此外， rsync 的传输方式至少可以透过三种方式来运作：

- 在本机上直接运作，用法就与 cp 几乎一模一样，例如： rsync -av /etc /tmp (将 /etc/ 的数据备份到 /tmp/etc 内)
- 透过 rsh 或 ssh 的信道在 server / client 之间进行数据传输，例如： rsync -av -e ssh user@rsh.server:/etc /tmp (将 rsh.server 的 /etc 备份到本地主机的 /tmp 内)
- 直接透过 rsync 提供的服务 (daemon) 来传输，此时 rsync 主机需要启动 873 port： 1. 你必须要在 server 端启动 rsync ， 看 /etc/xinetd.d/rsync 即可； 2. 你必须编辑 /etc/rsyncd.conf 配置文件； 3. 你必须设定好 client 端联机的密码数据； 4. 在 client 端可以利用：rsync -av user@hostname::/dir/path /local/path

其实三种传输模式差异在于有没有冒号 (:) 而已，本地端传输不需要冒号，透过 ssh 或 rsh 时，就得要利用一个冒号 (:)， 如果是透过 rsync daemon 的话，就得要两个冒号 (::) ，应该不难理解啦！因为本地端处理很简单， 而我们的系统本来就有提供 ssh 的服务，所以，底下鸟哥将直接介绍利用 rsync 透过 ssh 来备份的动作喔。 不过，在此之前咱们先来看看 rsync 的语法吧！

```
[root@www ~]# rsync [-avrlptgoD] [-e ssh] [user@host:/dir] [/local/path]
选项与参数：
-v ：观察模式，可以列出更多的信息，包括镜像时的档案档名等；
-q ：与 -v  相反，安静模式，略过正常信息，仅显示错误讯息；
-r ：递归复制！可以针对『目录』来处理！很重要！
-u ：仅更新 (update)，若目标档案较新，则保留新档案不会覆盖；
-l ：复制链接文件的属性，而非链接的目标源文件内容；
-p ：复制时，连同属性 (permission) 也保存不变！
-g ：保存源文件的拥有群组；
-o ：保存源文件的拥有人；
-D ：保存源文件的装置属性 (device)
-t ：保存源文件的时间参数；
-I ：忽略更新时间 (mtime) 的属性，档案比对上会比较快速；
-z ：在数据传输时，加上压缩的参数！
-e ：使用的信道协议，例如使用 ssh 通道，则 -e ssh
-a ：相当于 -rlptgoD ，所以这个 -a 是最常用的参数了！
更多说明请参考 man rsync 的解说！

# 1\. 将 /etc 的数据备份到 /tmp 底下：
[root@www ~]# rsync -av /etc /tmp
....(前面省略)....
sent 21979554 bytes  received 25934 bytes  4000997.82 bytes/sec
total size is 21877999  speedup is 0.99
[root@www ~]# ll -d /tmp/etc /etc
drwxr-xr-x. 106 root root 12288 Jul 26 16:10 /etc
drwxr-xr-x. 106 root root 12288 Jul 26 16:10 /tmp/etc &lt;==瞧！两个目录一样！
# 第一次运作时会花比较久的时间，因为首次建立嘛！如果再次备份呢？

[root@www ~]# rsync -av /etc /tmp
sent 55716 bytes  received 240 bytes  111912.00 bytes/sec
total size is 21877999  speedup is 390.99
# 比较一下两次 rsync 的传输与接受数据量，你就会发现立刻就跑完了！
# 传输的数据也很少！因为再次比对，仅有差异的档案会被复制。

# 2\. 利用 student 的身份登入 clientlinux.centos.vbird 将家目录复制到本机 /tmp
[root@www ~]# rsync -av -e ssh student@192.168.100.10:~ /tmp
student@192.168.100.10's password:  &lt;==输入对方主机的 student 密码
receiving file list ... done
student/
student/.bash_logout
....(中间省略)....
sent 110 bytes  received 697 bytes  124.15 bytes/sec
total size is 333  speedup is 0.41

[root@www ~]# ll -d /tmp/student
drwx------. 4 student student 4096 Jul 26 16:52 /tmp/student
# 瞧！这样就做好备份啦！很简单吧！
```

你可以利用上面的范例二来做为备份 script 的参考！不过要注意的是，因为 rsync 是透过 ssh 来传输数据的，所以你可以针对 student 这个家伙制作出免用密码登入的 ssh 密钥！ 如此一来往后异地备援系统就能够自动的以 crontab 来进行备份了！简单到爆！

免密码的 ssh 账号我们在上头已经讲过了，撰写 shell script 的能力也是必须要有的！利用 rsync 来进行你的备份工作吧！ ^_^！至于更多的 rsync 用法可以参考本章后面所列出的参考网站(注10)喔！

例题：在 clientlinux.centos.vbird (192.168.100.10) 上面，使用 vbirdtsai 的身份建立一只脚本，这只脚本可以在每天的 2:00am 主动的以 rsync 配合 ssh 取得 www.centos.vbird (192.168.100.254) 的 /etc, /root, /home 三个目录的镜像到 clientlinux.centos.vbird 的 /backups/ 底下。答：由于必须要透过 ssh 通道，且必须要使用 crontab 例行工作排程，因此

肯定要使用密钥系统的免密码账号。我们在 11.2.6 小节已经谈过相关作法， vbirdtsai 已经有了公钥与私钥档案，因此不要再使用 ssh-keygen 了，直接将公钥档案复制到 www.centos.vbird 的 /root/.ssh/ 底下即可。 实际作法可以是这样的：

```
# 1\. 在 clientlinux.centos.vbird 将公钥档复制给 www.centos.vbird 的 root
[vbirdtsia@clientlinux ~]$ scp ~/.ssh/id_rsa.pub root@192.168.100.254:~

# 2\. 在 www.centos.vbird 上面用 root 建置好 authorized_keys
[root@www ~]# ls -ld id_rsa.pub .ssh
-rw-r--r--. 1 root root  416 Jul 26 16:59 id_rsa.pub &lt;==有公钥档
drwx------. 2 root root 4096 Jul 25 11:44 .ssh       &lt;==有 ssh 的相关目录

[root@www ~]# cat id_rsa.pub &gt;&gt; ~/.ssh/authorized_keys
[root@www ~]# chmod 644 ~/.ssh/authorized_keys

# 3\. 在 clientlinux.centos.vbird 上面撰写 script 并测试执行：
[vbirdtsai@clientlinux ~]$ mkdir ~/bin ; vim ~/bin/backup_www.sh
#!/bin/bash
localdir=/backups
remotedir="/etc /root /home"
remoteip="192.168.100.254"

[ -d ${localdir} ] &#124;&#124; mkdir ${localdir}
for dir in ${remotedir}
do
        rsync -av -e ssh root@${remoteip}:${dir} ${localdir}
done

[vbirdtsai@clientlinux ~]$ chmod 755 ~/bin/backup_www.sh
[vbirdtsai@clientlinux ~]$ ~/bin/backup_www.sh
# 上面在测试啦！第一次测试可能会失败，因为鸟哥忘记 /backups 需要 root
# 的权限才能够建立。所以，请您再以 root 的身份去 mkdir 及 setfacl 吧！

# 4\. 建立 crontab 工作
[vbirdtsai@clientlinux ~]$ crontab -e
0 2 * * * /home/vbirdtsai/bin/backup_www.sh
```

11.6.3 透过 ssh 通道加密原本无加密的服务

现在我们知道 ssh 这个通道可以加密，而且，我们更知道 rsync 默认已经可以透过 ssh 通道来进行加密以进行镜像传输。 既然如此，那么其他的服务能不能透过这个 ssh 进行数据加密来传送信息呢？当然可以！很棒呢这个功能！ 要介绍实做之前，我们先用图示来谈一下作法。

假设服务器上面有启动了 VNC 服务在 port 5901 ，客户端则使用 vncviewer 要联机到服务器上的 port 5901 就是了。 那现在我们在客户端计算机上面启动一个 5911 的埠口，然后再透过本地端的 ssh 联机到服务器的 sshd 去，而服务器的 sshd 再去连接服务器的 VNC port 5901 。整个联机的图示如下所示：

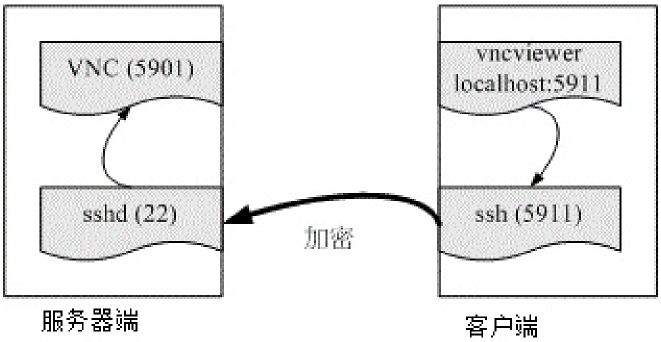


图 11.6-1、透过本地端的 ssh 加密联机到远程的服务器示意图

假设你已经透过上述各个小节建立好服务器 (www.centos.vbird) 上面的 VNC port 5901 ，而客户端则没有启动任何的 VNC 埠口。 那么你该如何透过 ssh 来进行加密呢？很简单，你可以在客户端计算机 (clientlinux.centos.vbird) 执行底下的指令：

```
[root@clientlinux ~]# ssh -L 本地埠口:127.0.0.1:远程端口 [-N] 远程主机
选项与参数：
-N ：仅启动联机通道，不登入远程 sshd 服务器
本地埠口：就是开启 127.0.0.1 上面一个监听的埠口
远程埠口：指定联机到后面远程主机的 sshd 后，sshd 该连到哪个埠口进行传输

# 1\. 在客户端启动所需要的端口进行的指令
[root@clientlinux ~]# ssh -L 5911:127.0.0.1:5901 -N 192.168.100.254
root@192.168.100.254's password:
   &lt;==登入远程仅是开启一个监听埠口，所以停止不能动作

# 2\. 在客户端在另一个终端机测试看看，这个动作不需要作，只是查阅而已
[root@clientlinux ~]# netstat -tnlp&#124; grep ssh
tcp  0   0 0.0.0.0:22           0.0.0.0:*             LISTEN      1330/sshd
tcp  0   0 127.0.0.1:5911       0.0.0.0:*             LISTEN      3347/ssh
tcp  0   0 :::22                :::*                  LISTEN      1330/sshd
[root@clientlinux ~]# netstat -tnap&#124; grep ssh
tcp  0   0 192.168.100.10:55490 192.168.100.254:22   ESTABLISHED 3347/ssh
# 在客户端启动 5911 的埠口是 ssh 启动的，同一个 PID 也联机到远程喔！
```

接下来你就可以在客户端 (192.168.100.10, clientlinux.centos.vbird) 使用『 vncviewer localhost:5911 』来联机， 但是该联机却会连到 www.centos.vbird (192.168.100.254) 那部主机的 port 5901 喔！不相信吗？ 当你达成 VNC 联机后，到 www.centos.vbird 那部主机上面瞧瞧就知道了：

```
# 3\. 在服务器端测试看看，这个动作不需要作，只是查阅而已
[root@www ~]# netstat -tnp &#124; grep ssh
tcp   0  0 127.0.0.1:59442     127.0.0.1:5901        ESTABLISHED 7623/sshd: root
tcp   0  0 192.168.100.254:22  192.168.100.10:55490  ESTABLISHED 7623/sshd: root
# 明显的看到 port 22 的程序同时联机到 port 5901 喔！
```

那如何取消这个联机呢？先关闭 VNC 之后，然后再将 clientlinux.centos.vbird 的第一个动作 (ssh -L ...) 按下 [ctrl]-c 就中断这个加密通道啰！这样会使用了吗？你可以将这个动作用在任何服务上喔！

11.6.4 以 ssh 信道配合 X server 传递图形接口

从前一个小节我们知道 ssh 可以进行程序的加密传递，亦即 ssh 通道啦！那么可不可以用在 X 上面呢？ 意思是说，那我能不能不要启动甚么很复杂的接口，就是在原有的接口底下使用 ssh 信道，将我所需要的服务器上面的图形接口传过来就好了？ 是可以的喔！鸟哥用一个 Windows 上面的 Xming X server 作范例好了。整个动作是这样的：

- 先在 Windows 上面启动 XLaunch，并设定好联机到 www.centos.vbird 的相关信息；
- 启动 Xming 程序，会取得一个 xterm 程序，该程序是 www.centos.vbird 的程序；
- 开始在 xterm 上面执行 X 软件，就会在 Windows 桌面上面显示啰！

那我们就开始来处理一下 Xming 这个程序吧！启动 XLaunch 之后出现下图模样：

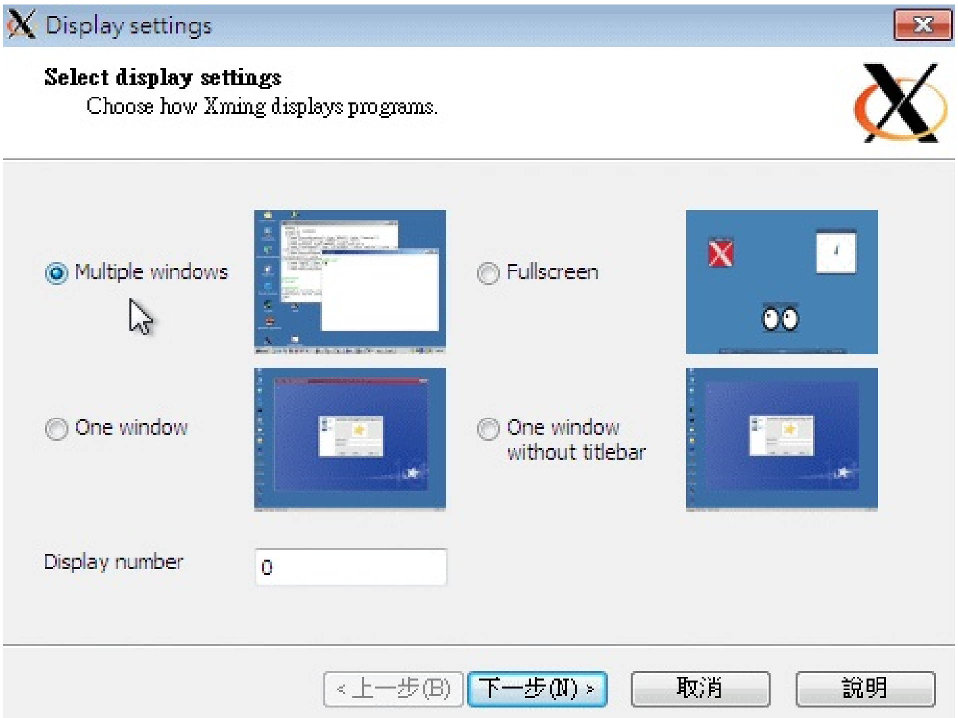


图 11.6-2、启动 XLaunch 程序-选择显示模式

记得上图中要选择 Multiple windows 会比较漂亮喔！然后按下『下一步』会出现下图：

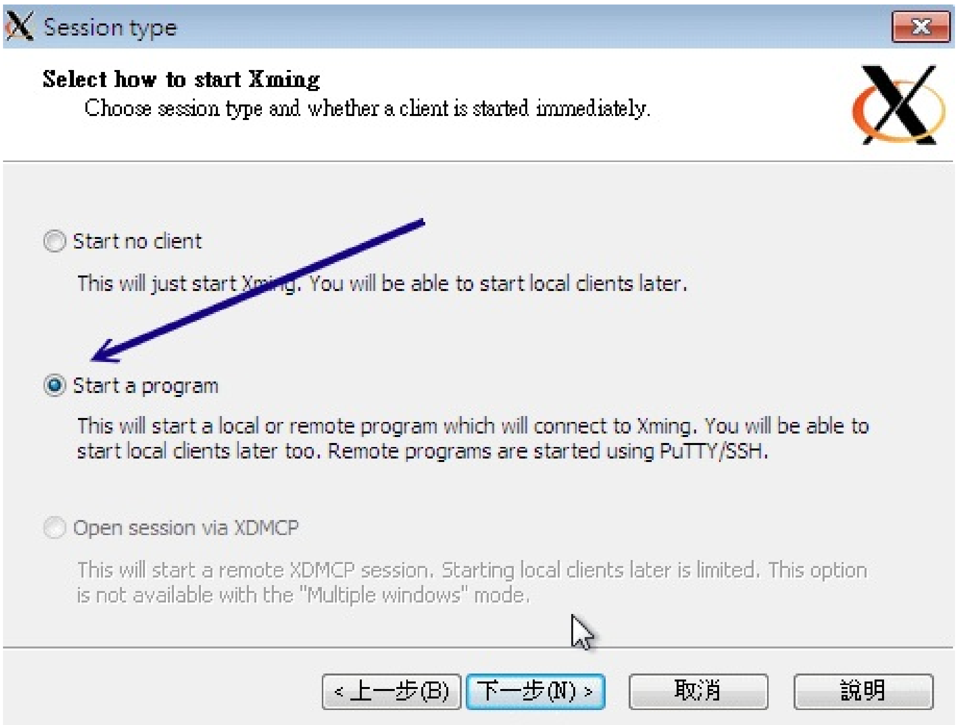


图 11.6-3、设定 XLaunch 程序-选择联机方式

我们要启动一只程序，并且是开放在 ssh/putty 之类的软件帮忙进行 ssh 信道的建立喔！然后下一步吧。

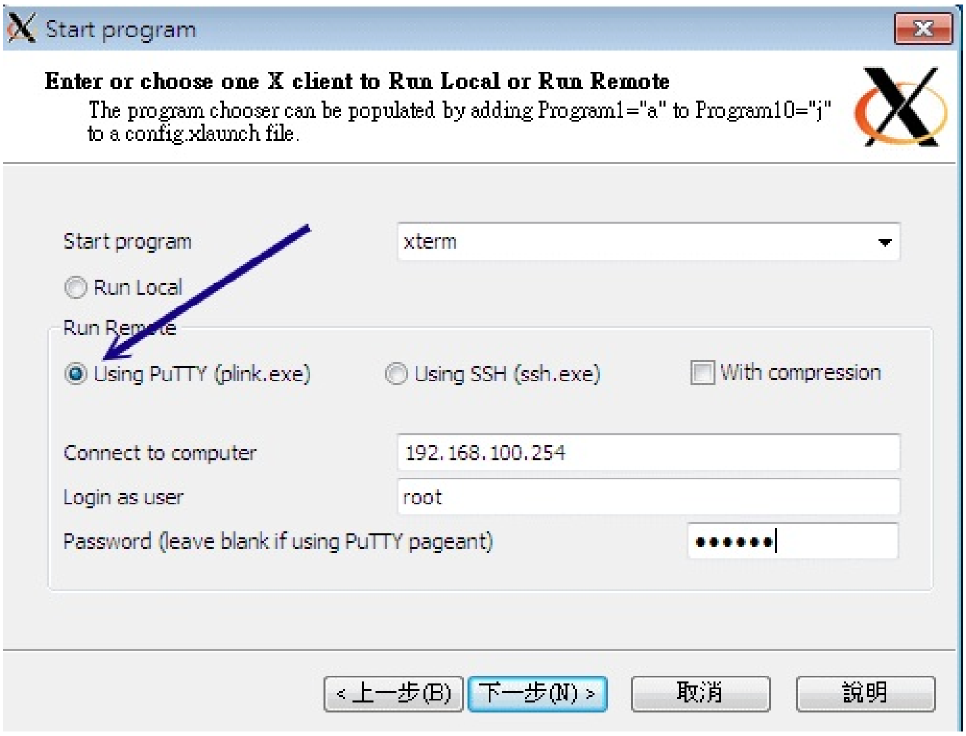


图 11.6-4、设定 XLaunch 程序-设定远程联机的相关参数

Xming 会主动的启动一个 putty 的程序帮你连进 sshd 服务器，所以这里得要帮忙设定好账号密码的相关信息。 鸟哥这里假设你的 sshd 尚未取消 root 登入，因此这里使用 root 的权限喔！

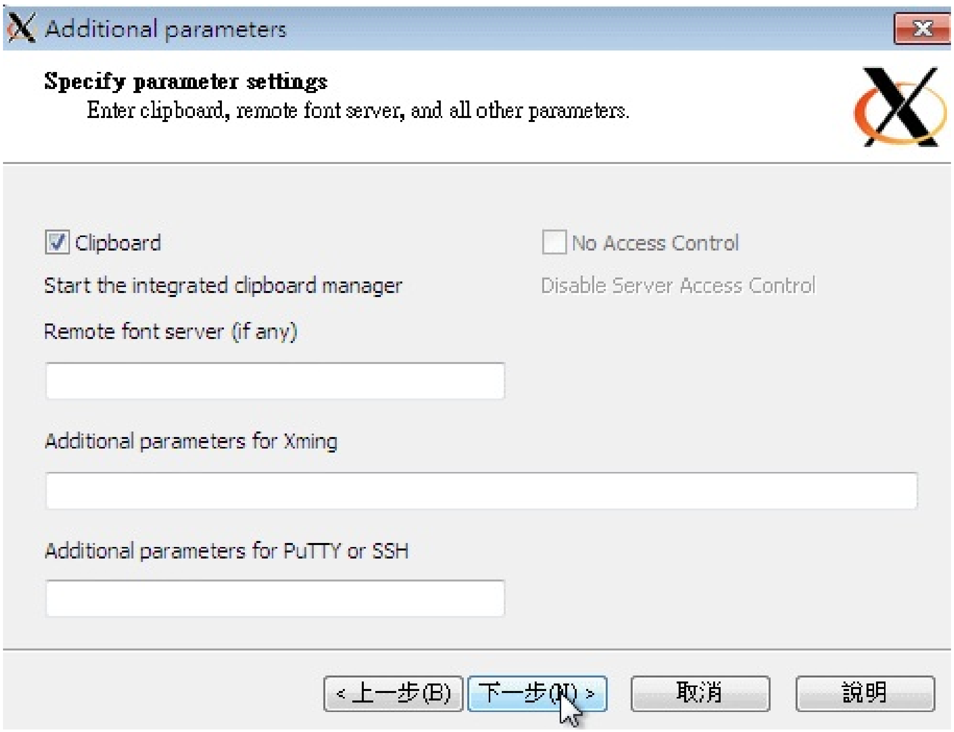


图 11.6-5、设定 XLaunch 程序-是否支持复制贴上功能

使用默认值吧！直接下一步。

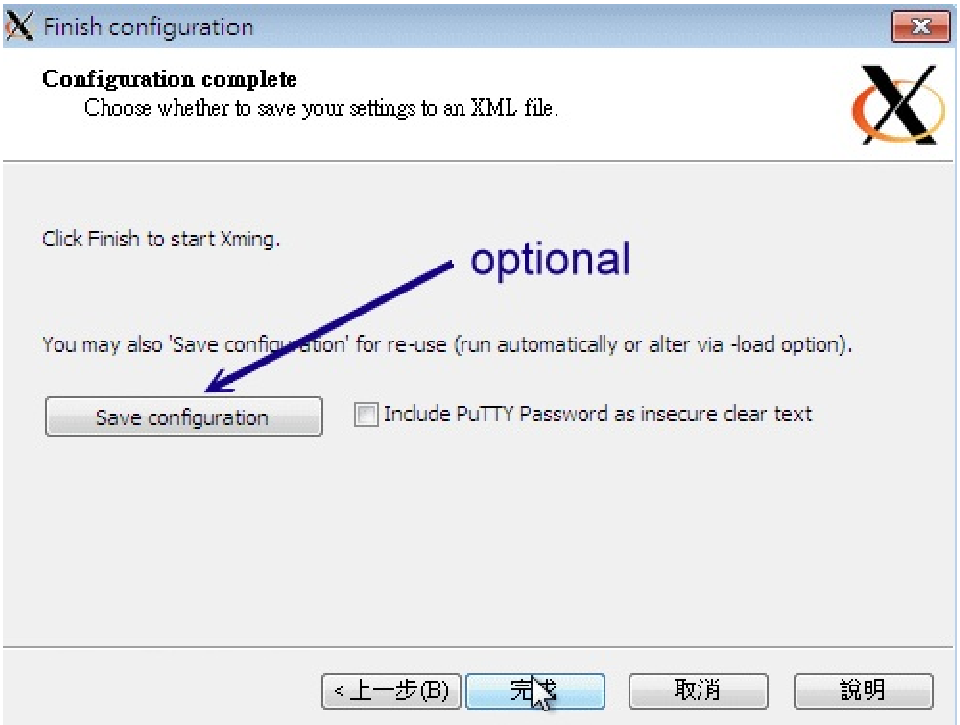


图 11.6-6、设定 XLaunch 程序-完成设定

很简单！这样就完成设定了！请按下完成，你就会看到 Windows 的桌面竟然出现如下的图示了！

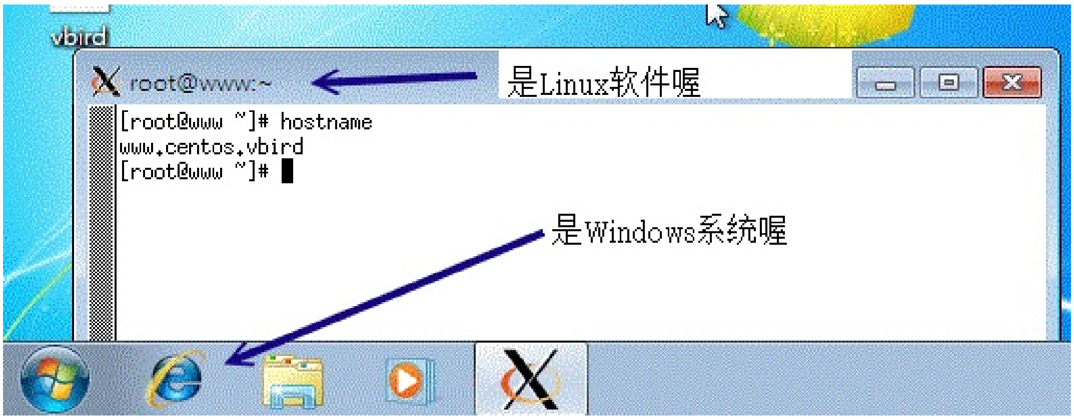


图 11.6-7、Windows 桌面出现的 X client 程序

上面这只程序就是 xterm 这个 X 的终端机程序。你可以在上面输入指令，该指令会传送到 Linux server ， 然后再将你要执行的图形数据透过 ssh 信道传送到目前的 Windows 上面的 Xming ，你的 Linux 完全不用启动 VNC, X, xrdp 等服务！只要有 sshd 就搞定了！就是这么简单！例如鸟哥输入几个游戏程序， 你的 Windows 窗口 (看任务栏就知道了) 就会出现这样的情况：

Tips: 事实上，我们的 basic server 安装方式并没有帮你安装 xterm 喔！所以，你得要自己安装 xterm 才行！ yum install xterm 就安装好啦！然后上面的动作再重来一次，就可以成功啰！而底下的图标里面的相关软件， 也是需要你自己安装的哟！ ^_^





图 11.6-8、Windows 桌面出现的 X client 程序

11.7 重点回顾

- 远程联机服务器可以让使用者在任何一部计算机登入主机，以使用主机的资源或管理与维护主机；
- 常见的远程登录服务有 rsh, telnet, ssh, vnc, xdmcp 及 RDP 等；
- telnet 与 rsh 都是以明码传输数据，当数据在 Internet 上面传输时较不安全；
- ssh 由于使用密钥系统，因此数据在 Internet 上面传输时是加密过的，所以较为安全；
- 但 ssh 还是属于比较危险的服务，请不要对整个 Internet 开放 ssh 的可登入权限，可利用 iptables 规范可登入范围；
- ssh 的 public Key 是放在服务器端，而 private key 是放在 client 端；
- ssh 的联机机制有两种版本，建议使用可确认联机正确性的 version 2 ；
- 使用 ssh 时，尽量使用类似 email 的方式来登入，亦即： ssh username@hostname
- client 端可以比对 server 传来的 public key 的一致性，利用的档案为 ~user/.ssh/known_hosts；
- ssh 的 client 端软件提供 ssh, scp, sftp 等程序；
- 制作不需要密码的 ssh 账号可利用 ssh-keygen -t rsa 来制作 public, private Key pair；
- 上述指令所制作出的 public key 必须要上传到 server 的 ~user/.ssh/authorized_keys 档案中；
- Xdmcp 是透过 X display manager (xdm, gdm, kdm 等) 所提供的功能协议；
- 若 client 端为 Linux 时，需要在 X 环境下以 xhost 增加可连接到本机 X Server 的 IP 才行；
- 除了 Xdmcp 之外，我们可以利用 VNC 来进行 X 的远程登录架构；
- VNC 预设开的 port number 为 5900 开始，每个 port 仅允许一个联机；
- rsync 可透过 ssh 的服务通道或 rsync --daemon 的方式来联机传输，其主要功能可以透过类似镜像备份， 仅备份新的数据，因此传输备份速度相当快速！

11.8 本章习题

- Telnet 与 SSH 都是远程联机服务器，为何我们都会推荐使用 SSH 而避免使用 Telnet 呢？原因何在？因为 Telnet 除了使用『明码』传送数据外，本身 telnet 就是很容易被入侵的一个服务器，所以当然也就比较危险了。 至于 ssh 其实也不是很安全的！由台湾计算机危机处理小组的文件可以明显的发现 openssl + openssh 也是常常有漏洞在发布！不过，比起 telnet 来说，确实是稍微安全一些！
- 请尝试说明 SSH 在 Server 与 Client 端联机时的封包加密机制；利用 key pair 来达到加密的机制：Server 提供 Public Key 给 Client 端演算 Private key ，以提供封包传送时的加密、解密！
- 请问 SSH 的配置文件是哪一个？如果我要修改让 root 无法使用 SSH 联机进入我的 SSH 主机，应该如何设定？又，如果要让 badbird 这个用户无法登入 SSH 主机，该如何设定？SSH 配置文件档名为 sshd_config ，通常放置在 /etc/ssh/sshd_config 内；如果不想让 root 登入，可以修改 sshd_config 内的参数成为：『PermitRootLogin no 』，并重新启动 ssh 来设定！如果要让 badbird 使用者无法登入，同样在 sshd_config 里面设定为：『DenyUsers badbird』即可！
- 在 Linux 上，预设的 Telnet 与 SSH 服务器使用的埠口(port number)各为多少？telnet 与 ssh 的埠口分别是：23 与 22！请参考 /etc/services 喔！
- 如果发现我无法在 Client 端使用 ssh 程序登入我的 Linux 主机，但是 Linux 主机却一切正常，可能的原因为何？(防火墙、known_hosts...)无法登入的原因可能有很多，最好先查询一下 /var/log/messages 里面的错误讯息来判断，当然，还有其他可能的原因为：

  1. 被防火墙挡住了，请以 iptables -L -n 来察看，当然也要察看 /etc/hosts.deny；
  2. 可能由于主机重新启动过， public key 改变了，请修改你的 ~/ssh/known_hosts 里面的主机 IP ；
  3. 可能由于 /etc/ssh/sshd_config 里面的设定问题，导致你这个使用者无法使用；
  4. 在 /etc/passwd 里面，你的 user 不具有可以登入的 shell ；
  5. 其他因素(如账号密码过期等等)
- 既然 ssh 是比较安全的资料封包传送方式，那么我就可以在 Internet 上面开放我的 Linux 主机的 SSH 服务了吗？！请说明你选择的答案的原因！最好不要对 Internet 开放你的 SSH 服务，因为 SSH 的加密函式库使用的是 openssl ，一般 Linux distribution 使用的 SSH 则是 openssh ，这两个套件事实上仍有不少的漏洞被发布过，因此，最好不要对 Internet 开放，毕竟 SSH 对于主机的权限是很高的！

11.9 参考数据与延伸阅读

- 注1：与 SSH 服务器有关的两个重要官网 OpenSSH 官方网站：http://www.openssh.com/ OpenSSL 官方网站：http://www.openssl.org/
- 注2：与 putty 及 pietty 有关的网站 putty 官方网站：http://www.chiark.greenend.org.uk/~sgtatham/putty/ pietty 中文网站：http://ntu.csie.org/~piaip/pietty/
- 注3：XDMCP 维基百科：http://en.wikipedia.org/wiki/X*display_manager*(program_type)
- 注4：教你怎么设定 gdm 的 custom.conf http://www.idevelopment.info/data/Unix/Linux/LINUX_ConfiguringXDMCPRedHatLinux.shtml http://www.yolinux.com/TUTORIALS/GDM_XDMCP.html
- 注5：自由的 X server -- Xming： http://sourceforge.net/projects/xming/
- 注6：与 VNC 相关的资料 man vncserver, man Xvnc 使用 X 的 VNC Module：http://phorum.study-area.org/viewtopic.php?t=25713 http://fedoranews.org/tchung/vnc/03.shtml
- 注7：维基百科：http://en.wikipedia.org/wiki/Remote_Desktop_Protocol
- 注8：官网在 http://xrdp.sourceforge.net/
- 注9： Fedora 基金会提供的 Extra Packages for Enterprise Linux (EPEL) 计划：http://fedoraproject.org/wiki/EPEL
- 注10：rsync 的相关用法介绍： 酷学园：用 rsync 做备份：http://phorum.study-area.org/viewtopic.php?t=15553 ADJ 实验室的 rsync + SSH：http://www.adj.idv.tw/server/linux_rsync.php
- 公钥加密机制的维基百科解释：http://en.wikipedia.org/wiki/Public_Key_Cryptography

---

2002/11/14：第一次完成 2003/03/08：加入标头说明，与修改部分内容，例如 Telnet 服务器软件的安装等等，以及 SSH 的 putty 使用中文状态！ 2003/09/09：将本文进行一些修订，此外，加入了课后练习！ 2005/07/02：将旧的文章移动到 这里 。 2005/07/07：好不容易将 VNC 还有 XDMCP 给他写了写～大家帮鸟哥参考看看啊～ 2005/07/09：加入了让 VNC 与 tty7 同步的 vnc.so 模块的说明 2005/11/22：加入了 RSH 服务器 的相关数据！ 2006/09/18：将 putty 的介绍转成 pietty 的介绍！因为 pietty 更好用！另外也将 rsh 重新改写一下，校稿过！ 2006/09/19：加入 rsync 的简易说明与操作！最文末的习题可以瞧一瞧！ 2011/02/15：将旧的基于 CentOS 4.x 的文章移动到 此处 2011/02/17：忍痛删除 telnet 服务器，毕竟真的很少用了～包括那个 rsh 也不再介绍！有兴趣的请参考 CentOS 4.x 的旧文章吧 2011/02/20：将 sshd 服务器作个简单的修改了，增加一些篇幅来说明相关例题与实做，尤其是 ~/.ssh/authorized_keys 的权限 2011/02/23：修改了许多 Xdmcp, VNC 的设定与图示，最重要是加入 xrdp 的安装与使用 2011/02/24：加入 Xming 透过 X11 forward from ssh 的方式！ 2011/07/25：将基于 CentOS 5.x 的版本移动到此处 2011/07/26：将所有的图示以及相关的网

站 IP 通通改为 CentOS 6.x 以及第三章谈到的区网架构啰！ 2011/11/24：经由网友的回报，在 ssh 联机方式里面谈到的公私钥系统是错误的！经过查询后，已经将正确的版本放上去了！ 2011/11/24：由于是较大幅度的改版，所以旧版也将它保留下来，按这里联机。

第十二章、网络参数控管者：DHCP 服务器

最近更新日期：2011/07/27

想象两种情况：(1)如果你在工作单位使用的是笔记本电脑，而且常常要带着你的笔记本电脑到处跑， 那么由第四章、连上 Internet 的说明中会发现，哇！ 我的网络卡参数要常常修改啊！而且，每到一个新的地方，就得问清楚该地的网络参数才行！真是麻烦。 (2)你的公司常常有访客或贵客来临，因为他们也带来笔电，所以也得常常跑来找你问网络参数才能设定他的计算机。 哇！这两种情况都会让你想哭哭吧？这个时候，动态主机设定协议 (DHCP) 可就大大的派上用场啦！DHCP 这个服务可以自动的分配 IP 与相关的网络参数给客户端，来提供客户端自动以服务器提供的参数来设定他们的网络。 如此一来，使用者只要将自己的笔电设定好经由 DHCP 协议来取得网络参数后，一插上网络线，呵呵！马上就可以享受 Internet 的服务啦！很方便吧！所以得来瞧一瞧这个好用的协定喔！

12.1 DHCP 运作的原理

在正式的进入 DHCP (Dynamic Host Configuration Protocol) 服务器设定之前，我们先来认识一下 DHCP 这个协议吧！还有，需要了解的是，我们是否『一定』得设定 DHCP 这个服务器呢？这里都需要厘清一下概念喔！

12.1.1 DHCP 服务器的用途

在开始 DHCP 的说明之前，我们先来复习一下之前在第二章网络基础里面提到的几个网络参数吧！ 要设定好一个网络的环境，使计算机可以顺利的连上 Internet ，那么你的计算机里面一定要有底下几个网络的参数才行，分别是：

- IP, netmask, network, broadcast, gateway, DNS IP

其中，那个 IP, netmask, network, broadcast 与 gateway 都可以在 /etc/sysconfig/network-scripts/ifcfg-eth[0-n] 这档案里面设定，DNS 服务器的地址则是在 /etc/resolv.conf 里头设定。只要这几个项目设定正确，那么计算机应该就没问题的可以上网了！ 所以说，你家里面的 3, 4 部计算机，你都可以手动的来设定好你所需要的网络参数， 然后利用 NAT 服务器的功能，就可以大摇大摆的连上 Internet 了！真是不错 ^_^，不是吗？

好了，现在让我们换一个大一些些的场景吧！假设你是学校宿舍的网络管理员，所管理的学生计算机大概有 100 部好了，那么你怎么设定好这 100 部的计算机呢？

1. 直接每一部计算机都让你登门拜访手动的去设定好？
2. 将所有的学生都集合起来，然后精神训话.....喔不！是直接教导一下怎么设定？还是
3. 藉由一部主机来自动的分配所有的网络参数给宿舍内的任何一部计算机？

这三种解决方案所需要的时间都不相同，如果你选择的是(1)，那么鸟哥个人认为，你不是工作狂就是疯掉了， 因为所要花费的时间与你所得的薪水与付出的心力是完全不成比例的。如果选择是(2)，那么很可能你会被挂上独裁者、 没良心的管理员的称号！如果是选择(3)呢？恭喜你！这个方案的管理时间花费最短，也是最不麻烦的作法啦！

呵呵！知道鸟哥要说些什么了吗？是的！这个 DHCP (Dynamic Host Configuration Protocol) 服务器最主要的工作，就是在进行上面的第三个方案，也就是自动的将网络参数正确的分配给网域中的每部计算机， 让客户端的计算机可以在开机的时候就立即自动的设定好网络的参数值，这些参数值可以包括了 IP、netmask、network、gateway 与 DNS 的地址等等。如此一来，身为管理员的你，只要注意到这一部提供网络参数的主机有没有挂掉就好了， 其他同学们的个人计算机，哈！你想都不必想要怎么去帮忙！因为 DHCP 主机已经完全都帮你搞定啦！ ^_^！ 阿！当管理员最大的幸福就是可以喝喝茶、聊聊天就能控管好一切的网络问题呢！

12.1.2 DHCP 协议的运作方式

你必需要知道的是，DHCP 通常是用于局域网络内的一个通讯协议，他主要藉由客户端传送广播封包给整个物理网段内的所有主机， 若局域网络内有 DHCP 服务器时，才会响应客户端的 IP 参数要求。所以啰，DHCP 服务器与客户端是应该要在同一个物理网段内的。 至于整个 DHCP 封包在服务器与客户端的来来回回情况有点像底下这样：(注1)

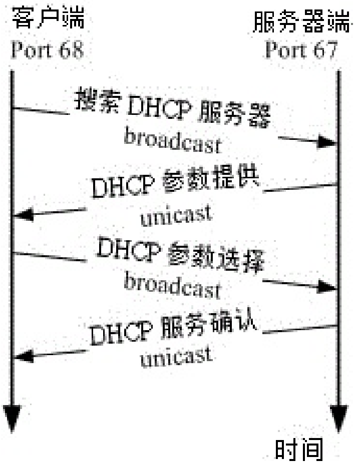


图 12.1-1、

户端取得 IP 参数的程序可以简化如下：

1. 客户端：利用广播封包发送搜索 DHCP 服务器的封包：

   若客户端网络设定使用 DHCP 协议取得 IP (在 Windows 内为『自动取得 IP』)，则当客户端开机或者是重新启动网络卡时， 客户端主机会发送出搜寻 DHCP 服务器的 UDP 封包给所有物理网段内的计算机。此封包的目标 IP 会是 255.255.255.255， 所以一般主机接收到这个封包后会直接予以丢弃，但若局域网络内有 DHCP 服务器时，则会开始进行后续行为。

2. 服务器端：提供客户端网络相关的租约以供选择：

   DHCP 服务器在接收到这个客户端的要求后，会针对这个客户端的硬件地址 (MAC) 与本身的设定数据来进行下列工作：

   - 到服务器的登录文件中寻找该用户之前是否曾经用过某个 IP ，若有且该 IP 目前无人使用，则提供此 IP 给客户端；
   - 若配置文件针对该 MAC 提供额外的固定 IP (static IP) 时，则提供该固定 IP 给客户端；
   - 若不符合上述两个条件，则随机取用目前没有被使用的 IP 参数给客户端，并记录下来。 总之，服务器端会针对客户端的要求提供一组网络参数租约给客户端选择，由于此时客户端尚未有 IP ，因此服务器端响应的封包信息中， 主要是针对客户端的 MAC 来给予回应。此时服务器端会保留这个租约然后开始等待客户端的回应。

3. 客户端：决定选择的 DHCP 服务器提供的网络参数租约并回报服务器：

由于局域网络内可能并非仅有一部 DHCP 服务器，但客户端仅能接受一组网络参数的租约。 因此客户端必需要选择是否要认可该服务器提供的相关网络参数的租约。当决定好使用此服务器的网络参数租约后， 客户端便开始使用这组网络参数来设定自己的网络环境。此外，客户端也会发送一个广播封包给所有物理网段内的主机， 告知已经接受该服务器的租约。此时若有第二台以上的 DHCP 服务器，则这些没有被接受的服务器会收回该 IP 租约。至于被接受的 DHCP 服务器会继续进行底下的动作。

4. 服务器端：记录该次租约行为并回报客户端已确认的响应封包信息：

   当服务器端收到客户端的确认选择后，服务器会回传确认的响应封包，并且告知客户端这个网络参数租约的期限， 并且开始租约计时喔！那么该次租约何时会到期而被解约 (真可怕的字眼) ？你可以这样想：

   - 客户端脱机：不论是关闭网络接口 (ifdown)、重新启动 (reboot)、关机 (shutdown) 等行为，皆算是脱机状态，这个时候 Server 端就会将该 IP 回收，并放到 Server 自己的备用区中，等待未来的使用；
   - 客户端租约到期：前面提到 DHCP server 端发放的 IP 有使用的期限，客户端使用这个 IP 到达期限规定的时间，而且没有重新提出 DHCP 的申请时，就需要将 IP 缴回去！这个时候就会造成断线。但用户也可以再向 DHCP 服务器要求再次分配 IP 啰。

以上就是 DHCP 这个协议在 Server 端与 Client 端的运作状态，由上面这个运作状态来看，我们可以晓得，只要 Server 端设定没有问题，加上 Server 与 Client 在硬件联机上面确定是 OK 的，那么 Client 就可以直接藉由 Server 来取得上网的网络参数，当然啦，只要我们这些管理员能够好好的、正确的管理好我们的 DHCP 服务器， 嘿嘿！那么上网的设定自然就变成一件很简单的事情啦！不过，关于上述的流程还是有一些需要额外说明的啦：

- DHCP 服务器给予客户端的 IP 参数为固定或动态：

在上面的步骤里面，注意到第二步骤了吗？就是服务器会去比较客户端的 MAC 硬件地址，并判断该 MAC 是否需要给予一个固定的 IP 呢！所以啦，我们可以设定 DHCP 服务器给予客户端的 IP 参数主要有两种：

- 固定 (Static) IP：

  只要那个客户端计算机的网络卡不换掉，那么 MAC 肯定就不会改变，由于 DHCP 可以根据 MAC 来给予固定的 IP 参数租约，所以该计算机每次都能以一个固定的 IP 连上 Internet ！呵呵！ 这种情况比较适合当这部客户端计算机需要用来做为提供区域内的一些网络服务的主机之用 (所以 IP 要固定)。那么如何在 Linux 上面知道网络卡的 MAC 呢？很简单啦！有很多的方式，最简单的方式就是使用 ifconfig 及 arp 来进行：

```
# 1\. 观察自己的 MAC 可用 ifconfig：
[root@www ~]# ifconfig &#124; grep HW
eth0      Link encap:Ethernet  HWaddr 08:00:27:71:85:BD
eth1      Link encap:Ethernet  HWaddr 08:00:27:2A:30:14
# 因为鸟哥有两张网卡，所以有两个硬件地址喔！

# 2\. 观察别人的 MAC 可用 ping 配合 arp
[root@www ~]# ping -c 3 192.168.1.254
[root@www ~]# arp -n
Address          HWtype  HWaddress           Flags Mask    Iface
192.168.1.254    ether   00:0c:6e:85:d5:69   C             eth0
```

- 动态 (dynamic) IP：

  Client 端每次连上 DHCP 服务器所取得的 IP 都不是固定的！都直接经由 DHCP 所随机由尚未被使用的 IP 中提供！

除非你的局域网络内的计算机有可能用来做为主机之用，所以必需要设定成为固定 IP ，否则使用动态 IP 的设定比较简单，而且使用上面具有较佳的弹性。怎么说呢？假如你是一个 ISP 好了，而你只申请到 150 个 IP 来做为你的客户联机之用。那么你是否真的只能邀集到 150 的使用者？呵呵！当然不啰！我可以邀集 200 个使用者以上呢！

为什么？这样想好了，我今天开了一家餐馆，里面只有 20 个座位，那么是否我一餐只能卖给 20 个人呢？当然不是啦！因为客人是人来人往的，有人先吃有人后吃，所以同样是 20 个座位，但是可以有 40 个人来吃我的简餐，因为来的时间不一样嘛！了解了吗？呵呵！对啦！你这个 ISP 虽然只有 150 个 IP 可以发放，但是因为你的使用者并非 24 小时都挂在在线的，所以你可以将这 150 个 IP 做良好的分配，让 200 个人来『轮流使用』这 150 个 IP 哩！

Tips: 其实 IP 只有 Public IP 与 Private IP 两种，中文翻译成『公共 IP』与『私有 IP』这两个， 至于其他所谓的『静态 IP』、『实体 IP』、『虚拟 IP』、『浮动式 IP』等等，都是藉由一些 IP 取得的方式来分类的， 关于 IP 的种类我们在网络基础中谈过了， 记得再好好的厘清一下观念喔！



事实上现在主流的 ADSL 宽带拨接上网也有使用到『静态 IP 』与『固定 IP 』之类的概念喔！ 举例来说好了，hinet/seed net 等主要 ISP 都有提供所谓的：『一个固定 IP 搭配 7~8 个浮动 IP 』的 ADSL 拨接功能，也就是说同样透过一条电话线拨接到 ISP ，但是其中一个拨接是可以取得固定的 IP 呢！ 而其他的则是非固定的 IP ，DHCP 的 static/dynamic 跟这个玩意儿有点类似啦！ ^_^

- 关于租约所造成的问题与租约期限：

怪了！如果我们观察上面 DHCP 运作模式的第四个步骤，你会发现最后 DHCP 服务器还会给予一个租约期限！ 干嘛还要这样的一个期限呢？其实设定期限还是有个优点啦！最大的优点就是可以避免 IP 被某些使用者一直占用着，但该使用者却是 Idle (发呆) 的状态！

举个例子来说，我们刚刚不是说到，我有 150 个 IP ，但是偏偏我有 200 个用户吗？我们以 2010 年的世界杯足球赛来说明好了。假设每个使用者都急着上网知道世足赛的消息， 那么某些热门对战时段网络将可能达到使用尖峰！也就是说，这 200 个人同时要来使用这 150 个 IP ，有可能吗？当然不可能！肯定会有 50 个人无法联机，因为『很抱歉！目前系统正在忙线中，请你稍后再拨！』

那怎么办？这个时候租约到期的方式就很有用处啦！那几个已经联机进来很久的人， 就会因为租约到期而被迫脱机，这个时候该 IP 就会被释放出来，哈哈！大家赶快抢呀！先抢到先赢喔！ 所以，那 50 个人 (包括被迫脱机的那个朋友) 只好继续的、努力的、加油的来进行 DHCP 的要求啰！ ^_^""

虽然说是优点，但是其实如果站在使用者的角度来看，还是可能会造成公愤的！凭什么大家一起交钱， 我先联机进来就需要先被踢出去？～呵呵！所以啰，如果要当 ISP ，还是得要先规划好服务的方针才行呦！ 这样你可以了解租约到期的行为了吗？！ ^_^

既然有租约时间，那么是否代表我用 DHCP 取得的 IP 就得要『手动』的在某个时间点去重新取得新的 IP 呢？不需要的啦！因为目前的 DHCP 客户端程序大多会主动的依据租约时间去重新申请 IP (renew) 的！也就是说在租约到期前你的 DHCP 客户端程序就已经又重新申请更新租约时间了。所以除非 DHCP 主机挂点， 否则你所取得的 IP 应该是可以一直使用下去的！

Tips: 一般来说，假设租约期限是 T 小时，那么客户端在 0.5T 会主动向 DHCP 服务器发出重新要求网络参数的封包。 如果这次封包要求没有成功，那么在 0.875T 后还会再次的发送封包一次。正因如此，所以<u>服务器端会启动 port 67 监听用户要求，而用户会启动 port 68 主动向服务器要求</u>哩！鸟哥觉得这是很特殊的一件事呢！



- 多部 DHCP 服务器在同一物理网段的情况

或许你曾经发现过一件事情，那就是当我的网域里面有两部以上的 DHCP 服务器时， 到底哪一部服务器会提供我的这部客户端计算机所发出的 DHCP 要求？呵呵！很抱歉，俺也不晓得！ 因为在网络上面，很多时候都是『先抢先赢』的， DHCP 的回应也是如此！当 Server1 先响应时，你使用的就是 Server1 所提供的网络参数内容，如果是 Server2 先响应，你就是使用 Server2 的参数来设定你的客户端 PC ！不过，前提之下当然是这些计算机的『物理联机』都是在一起的啊！

因为这个特色的关系，所以当你在练习 DHCP 服务器的设定之前， 不要在已经正常运作的区网下测试，否则会很惨。举个鸟哥的例子来说好了，某一次其他系的研究生在测试网络安全时， 在原有的区网上面放了一部 IP 分享器，结果你猜怎么着？整栋大楼的网络都不通了！因为那时整栋大楼的网络是串接在一起的， 而我们学校是使用 DHCP 让客户端上网。由于 IP 分享器的设定并不能连上 Internet ，哇！大家都无法上网了啦！ 那你晓得了吗？不要随便测试啦这个 DHCP 服务器！

12.1.3 何时需要架设 DHCP 服务器

既然 DHCP 的好处是『免客户端设定』，而且对于行动装置的上网方面非常的方便！那么是否代表你就得要架设一部 DHCP 呢？那可不一定！接下来要告知大家的是几个概念性的问题， 你倒不一定『必需』遵守底下的一些概念呢！反正，自己的网域自己『爽』就好啦！

- 使用 DHCP 的几个时机

在某些情况之下，倒是强烈的建议架设 DHCP 主机的！什么情况呢？例如：

- 具有相当多行动装置的场合： 例如你的公司内部很多笔记本电脑使用的场合！因为这种笔电本身就是移动性的装置， 如果每到一个地方都要去问人家『喂！你这边的网络参数是什么？』还得要担心是否会跟人家的 IP 相冲突等等的问题！这个时候，DHCP 可就是你的救星啰！

- 区域内计算机数量相当的多时： 另外一个情况就是你所负责的网域内计算机数量相当庞大时， 大到你没有办法一个一个的进行说明来设定他们自己的网络参数，这个时候为了省麻烦，还是架设 DHCP 来的方便呐！况且，维护一部你熟悉的 DHCP 主机，要比造访几十个不懂计算机的人要简单的多哩！^_^

- 不建议使用 DHCP 主机的时机

虽然 DHCP 有很多好处，但是你有没有发现一个步骤怪怪的呀！回头看一下那个步骤一， 客户端在开机的时候会主动的发送讯息给网域上的所有机器，这个时候，如果网域上就是没有 DHCP 主机呢？很抱歉，那么你的这部客户端计算机，『仍然会持续的发送讯息！』 眞正的时间与次数不晓得会有多久，不过，肯定会超过 30 秒以上， 甚至可以达到一分钟以上！哇！那么这段时间你能干嘛？呵呵！除了等、还是等！ 所以啰，如果计算机数不多，还是使用手动的方式来设定一下就好了！方便嘛！

- 在你网域内的计算机，有很多机器其实是做为主机的用途，很少用户需求，那么似乎就没有必要架设 DHCP ；
- 更极端的情况是，像一般家里，只有 3 ~ 4 部计算机，这个时候，架设 DHCP 只能拿来练练功力，事实上，并没有多大的效益；
- 当你管理的网域当中，大多网络卡都属于老旧的型号，并不支持 DHCP 的协议时；
- 很多用户的信息知识都很高，那么也没有需要架设 DHCP 啦。

如前所述，上面的都是概念性的说法，事实上，一件事情的解决之道是有很多的方案的， 没有所谓的『完全正确』的方案，只有『相对可行、并且符合经济效益与功能』的方案！ 所以啰，架设任何网站之前，请先多评估评估呐！

12.2 DHCP 服务器端的设定

事实上，目前市面上的 IP 分享器已经便宜到爆了！而 IP 分享器本身就含有 DHCP 的功能。所以如果你只是想要单纯的使用 DHCP 在你的局域网络当中而已，那么建议你直接购买一部 IP 分享器来使用即可， 因为至少它很省电。如果你还有其他考虑的话，才来架设 DHCP 吧！底下我们以一个简单的范例来架设 DHCP 先。

12.2.1 所需软件与档案结构

DHCP 的软件需求很简单，就是只要服务器端软件即可，在 CentOS 6.x 上面，这个软件的名称就是 dhcp 啰！如果是默认安装，那么这个软件是不会安装的，请自行使用 yum 去装好这个软件吧！安装完毕之后， 你可以使用『 rpm -ql dhcp 』来看看这个软件提供了哪些档案，基本上，比较重要的档案数据如下：

- /etc/dhcp/dhcpd.conf 这个就是 dhcp 服务器的主要配置文件咯！在某些 Linux 版本上头这个档案可能不存在，所以如果你确定有安装 dhcp 软件却找不到这个档案时，请手动自行建立它即可。

  Tips: 其实 dhcp 软件在释出的时候都会附上一个范例档案，你可以使用『 rpm -ql dhcp 』来查询到 dhcpd.conf.sample 这个范例档案，然后将该档案复制成为 /etc/dhcp/dhcpd.conf 后，再手动去修改即可，这样设定比较容易啦！



- /usr/sbin/dhcpd 启动整个 dhcp daemon 的执行档啊！其实最详细的执行方式应该要使用『 man dhcpd 』来查阅一番的呢！^_^
- /var/lib/dhcp/dhcpd.leases 这档案颇有趣的！我们前面原理部分不是有提到『租约』吗？DHCP 服务器端与客户端租约建立的启始与到期日就是记录在这个档案当中的咯！

就跟你说很简单吧！整个软件数据也不过才如此而已呢！

12.2.2 主要配置文件 /etc/dhcp/dhcpd.conf 的语法

在 CentOS 5.x 以前，这个档案都被放置到 /etc/dhcpd.conf 的，新版的才放置于此处。其实 DHCP 的设定很简单啊，只要将 dhcpd.conf 设定好就可以启动了。不过编辑这个档案时你必须要留意底下的规范：

- 『 # 』为批注符号；
- 除了右括号 ")" 后面之外，其他的每一行设定最后都要以『 ; 』做为结尾！重要！
- 设定项目语法主要为：『 <参数代号> <设定内容> 』，例如： default-lease-time 259200;
- 某些设定项目必须以 option 来设定，基本方式为『 option <参数代码> <设定内容> 』例如： option domain-name "your.domain.name";

基本上，我们刚刚前面提过说， DHCP 的 IP 分配可分为给予动态 IP 与固定 IP ，其中又需要了解的是， 如果需要设定固定 IP 的话，那么就必须要知道要设定成固定 IP 的那部计算机的硬件地址 (MAC) 才行，请使用 arp 或 ifconfig 来查知你的接口的 MAC 吧！好了，那么需要设定的项目有哪些呢？ 其实 dhcpd.conf 里头的设定主要分为两大项目，一个是服务器运作的整体设定 (Global) 一个是 IP 设定模式 (动态或固定)， 每个项目的设定值大概有底下这几项：

- 整体设定 (Global)

假设你的 dhcpd 只管理一个区段的区网，那么除了 IP 之外的许多网络参数就可以放在整体设定的区域中，这包括有租约期限、DNS 主机的 IP 地址、路由器的 IP 地址还有动态 DNS (DDNS) 更新的类型等等。当固定 IP 及动态 IP 内没有规范到某些设定时，则以整体设定值为准。这些参数的设定名称为：

- default-lease-time 时间： 用户的计算机也能够要求一段特定长度的租约时间，但若使用者没有特别要求租约时间的话， 那么就以此为预设的租约时间。后面的时间参数默认单位为秒；

- max-lease-time 时间： 与上面的预设租约时间类似，不过，这个设定值是在规范使用者所能要求的最大租约时间。也就是说， 使用者要求的租约时间若超过此设定值，则以此值为准；

- option domain-name "领域名"： 如果你在 /etc/resolv.conf 里面设定了一个『 search google.com 』的话，这表示当你要搜寻主机名时， DNS 系统会主动帮你加上这个领域名的意思。

- option domain-name-servers IP1, IP2： 这个设定参数可以修改客户端的 /etc/resolv.conf 档案！就是 nameserver 后面接的那个 DNS IP 啰！特别注意设定参数最末尾为『servers』 (有 s 喔)；

- ddns-update-style 类型： 因为 DHCP 客户端所取得的 IP 通常是一直变动的，所以某部主机的主机名与 IP 的对应就很难处理。此时 DHCP 可以透过 ddns (请参考第十章与第十九章 DNS 的说明) 来更新主机名与 IP 的对应。不过我们这里不谈这么复杂的东西，所以你可以将他设定为 none 喔。

- ignore client-updates： 与上一个设定值较相关，客户端可以透过 dhcpd 服务器来更新 DNS 相关的信息。不过，这里我们也先不谈这个， 因此就将它设定为 ignore (忽略) 了。
- option routers 路由器的地址： 设定路由器的 IP 所在，记得那个『 routers 』要加 s 才对！
- IP 设定模式 (动态或固定)

由于 dhcpd 主要是针对局域网络来给予 IP 参数的，因此在设定 IP 之前，我们得要指定一个区网才行。 指定区网的方式使用如下的参数：

subnet NETWORK_IP netmask NETMASK_IP { ... }

我们知道区网要给予 network / netmask IP 这两个参数才行，例如之前谈过的： 192.168.100.0 / 255.255.255.0 这样的设定值。 上头设定值当中， subnet 与 netmask 是关键词，而大写部分就填上你的区网参数啰。那在括号内还有什么参数需要设定的？ 那就是到底 IP 是固定的还是动态的设定啊：

- range IP1 IP2： 在这个区网当中，给予一个连续的 IP 群用来发放成动态 IP 的设定，那个 IP1 IP2 指的是开放的 IP 范围。 举例来说，你想要开放 192.168.100.101 到 192.168.100.200 这 100 个 IP 用来作为动态分配，那就是： range 192.168.100.101 192.168.100.200;
- host 主机名 { ... }; 这个 host 就是指定固定 IP 对应到固定 MAC 的设定值，那个主机名可以自己想想再给予即可。 不过在大括号内就得要指定 MAC 与固定的 IP 啰！那这两个设定值怎么设定呢？看看底下啰：
    - hardware ethernet 硬件地址： 利用网络卡上面的固定硬件地址来设定，亦即该设定仅针对这个硬件地址有效的意思；
    - fixed-address IP地址： 给予一个固定的 IP 地址的意思。

说再多也没有什么用啦！让我们实际来玩一个案例吧！你就知道该如何处理了。

12.2.3 一个局域网络的 DHCP 服务器设定案例

假设我的环境当中，Linux 主机除了 NAT 服务器之外还得要负责其他服务器，例如邮件服务器的支持。 而在后端局域网络中则想要提供 DHCP 的服务。整个硬件配置的情况就如同第三章的图 3.2-1所示的内部独立区网 (centos.vbird 网域)。 需要注意的是，在图中 Linux Router 有两块接口，其中 eth1 对内而 eth0 对外，至于其他的网络参数设计为：

- Linux 主机对内的 eth1 的 IP 设定为 192.168.100.254 这个；
- 内部网段设定为 192.168.100.0/24 这一段，且内部计算机的 router 为 192.168.100.254

，此外 DNS 主机的 IP 为中华电信的 168.95.1.1 及 Seednet 的 139.175.10.20 这两个；

- 我想要让每个使用者预设租约为 3 天，最长为 6 天；
- 只想要分配的 IP 只有 192.168.100.101 到 192.168.100.200 这几个，其他的 IP 则保留下来；
- 我还有一部主机，他的 MAC 是『 08:00:27:11:EB:C2 』，我要给他的主机名为 win7 ，且 IP 为 192.168.100.30 这个 (请对照图 3.2-1 喔！)。

那我的配置文件就会像底下这个样子了：

```
[root@www ~]# vim /etc/dhcp/dhcpd.conf
# 1\. 整体的环境设定
ddns-update-style            none;            &lt;==不要更新 DDNS 的设定
ignore client-updates;                        &lt;==忽略客户端的 DNS 更新功能
default-lease-time           259200;          &lt;==预设租约为 3 天
max-lease-time               518400;          &lt;==最大租约为 6 天
option routers               192.168.100.254; &lt;==这就是预设路由
option domain-name           "centos.vbird";  &lt;==给予一个领域名
option domain-name-servers   168.95.1.1, 139.175.10.20;
# 上面是 DNS 的 IP 设定，这个设定值会修改客户端的 /etc/resolv.conf 档案内容

# 2\. 关于动态分配的 IP
subnet 192.168.100.0 netmask 255.255.255.0 {
    range 192.168.100.101 192.168.100.200;  &lt;==分配的 IP 范围

    # 3\. 关于固定的 IP 啊！
    host win7 {
        hardware ethernet    08:00:27:11:EB:C2; &lt;==客户端网卡 MAC
        fixed-address        192.168.100.30;    &lt;==给予固定的 IP
    }
}
# 相关的设定参数意义，请查询前一小节的介绍，或者 man dhcpd.conf
```

够简单吧！这样就设定好了！你可以复制上头的数据然后修改一下，让里头的 IP 参数符合你的环境， 就能够设定好你的 DHCP 服务器了。接下来理论上你就能够启动 dhcp 了。不过，在某些早期的 Linux distribution 上面， 当你的 Linux 主机具有多个接口时，你的一个设定可能会让多个接口同时来监听，那就可能会发生错误了。

举例来说，我们现在的设定是 192.168.100.0/24 这个在 eth1 上头的网域，假设你还有一个界面 eth2 在 192.168.2.0/24 好了， 那万一你的 DHCP 同时监听两块接口的话，想一想，如果 192.168.2.0/24 网域的客户端发送出 dhcp 封包的要求时， 他会取得什么 IP ？当然是 192.168.100.X ！所以啰，我们就得要针对 dhcpd 这个执行文件设定他监听的接口， 而不是针对所有的接口都监听啊！你说是吧！^_^！那如何处理呢？在 CentOS (Red Hat 系统) 可以这样做：

```
[root@www ~]# vim /etc/sysconfig/dhcpd
DHCPDARGS="eth0"
```

不过这个动作在 CentOS 5.x 以后的版本上面已经不需要了，因为新版本的 dhcp 会主动的分析服务器的网段与实际的 dhcpd.conf 设定， 如果两者无法吻合，就会有错误提示，人性化多了。^_^！接下来我们可以开始启动 dhcp 试看看啰！

12.2.4 DHCP 服务器的启动与观察

开始来启动 dhcp 吧！在启动前你得要注意几件事情喔：

- 你的 Linux 服务器网络环境已经设定好，例如 eth1 已经是 192.168.100.254；
- 你的防火墙规则已经处理好，例如：(1)放行内部区网的联机、(2)iptables.rule 的 NAT 服务已经设定妥当；

另外你要注意的是：dhcpd 使用的埠口是 port 67 ，并且启动的结果会记录在 /var/log/messages 档案内，你最好能去观察一下 /var/log/messages 所显示的 dhcpd 相关信息才好。

```
# 1\. 启动后观察一下埠口的变化：
[root@www ~]# /etc/init.d/dhcpd start
[root@www ~]# chkconfig dhcpd on
[root@www ~]# netstat -tlunp &#124; grep dhcp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address Foreign Address PID/Program name
udp        0      0 0.0.0.0:67    0.0.0.0:*       1581/dhcpd

# 2\. 固定去看看登录文件的输出信息
[root@www ~]# tail -n 30 /var/log/messages
Jul 27 01:51:24 www dhcpd: Internet Systems Consortium DHCP Server 4.1.1-P1
Jul 27 01:51:24 www dhcpd: Copyright 2004-2010 Internet Systems Consortium.
Jul 27 01:51:24 www dhcpd: All rights reserved.
Jul 27 01:51:24 www dhcpd: For info, please visit https://www.isc.org/software/dhcp/
Jul 27 01:51:24 www dhcpd: WARNING: Host declarations are global.  They are not
limited to the scope you declared them in.
Jul 27 01:51:24 www dhcpd: Not searching LDAP since ldap-server, ldap-port and
ldap-base-dn were not specified in the config file
Jul 27 01:51:24 www dhcpd: Wrote 0 deleted host decls to leases file.
Jul 27 01:51:24 www dhcpd: Wrote 0 new dynamic host decls to leases file.
Jul 27 01:51:24 www dhcpd: Wrote 0 leases to leases file.
Jul 27 01:51:24 www dhcpd: Listening on LPF/eth1/08:00:27:2a:30:14/192.168.100.0/24
Jul 27 01:51:24 www dhcpd: Sending on   LPF/eth1/08:00:27:2a:30:14/192.168.100.0/24
....(以下省略)....
```

看到这些资料就是成功的象征啦！尤其是上述有特殊字体的部分。恭喜你啊！真是『福气啦！』不过， 万一你看到的登录档是类似底下的模样呢？

```
Jul 27 01:56:30 www dhcpd: /etc/dhcp/dhcpd.conf line 7: unknown option
dhcp.domain-name-server
Jul 27 01:56:30 www dhcpd: option domain-name-server#011168.
Jul 27 01:56:30 www dhcpd:          ^
Jul 27 01:56:30 www dhcpd: /etc/dhcp/dhcpd.conf line 9: Expecting netmask
Jul 27 01:56:30 www dhcpd: subnet 192.168.100.0 network
Jul 27 01:56:30 www dhcpd:                        ^
Jul 27 01:56:30 www dhcpd: Configuration file errors encountered -- exiting
```

上述的数据表示在第 7, 9 行恐怕有点设定错误，设定错误的地方在行号底下还有指数符号 (^) 特别标注出来！ 由上面的情况来看，第 7 行的地方应该是 domain-name-servers 忘了加 s 了，而第 9 行则是参数下错， 应该是 netmask 而非 network ！这样了解乎？

12.2.5 内部主机的 IP 对应

如果你有仔细的瞧过第二章的网络基础的话，那么应该还会记得那个 /etc/hosts (第四章 4.4.1) 会影响内部计算机在联机阶段的等待时间吧？那么我现在使用 DHCP 之后，糟糕！我怎么知道哪一部 PC 连上我的主机，那要怎么填写 /etc/hosts 的内容呢？这真是太简单了！就将所有可能的计算机 IP 都加进去该档案呀！ ^_^ ！以鸟哥为例，在这个例子中，鸟哥的分配的 IP 至少有 192.168.100.30, 192.168.100.101 ~ 192.168.100.200 ，所以 /etc/hosts 可以写成：

```
[root@www ~]# vim /etc/hosts
127.0.0.1    +    localhost.localdomain localhost
192.168.100.254   vbird-server
192.168.100.30    win7
192.168.100.101   dynamic-101
192.168.100.102   dynamic-102
....(中间省略)....
192.168.100.200   dynamic-200
```

这样一来，所有可能连进来的 IP 都已经有纪录了，哈哈！当然没有什么大问题啰！ ^_^！不过， 更好的解决方案则是架设内部的 DNS 服务器，这样一来，内部的其他 Linux 服务器也不必更改 /etc/hosts 就能够取得每部主机的 IP 与主机名对应，那样就更加妥当啦！

12.3 DHCP 客户端的设定

DHCP 的客户端可以是 Windows 也可以是 Linux 呢！鸟哥的网域内使用三部计算机，就如图 3.2-1 所示的那样。 Linux 与 Windows XP 的设定方式已经分别在第四章与第三章谈过了，底下就稍微介绍过而已。至于图示的部分， 我们主要是以 Windows 7 来做介绍啰。

12.3.1 客户端是 Linux

Linux 的网络参数设定还记得吧？不记得的话就得要打屁股了！在第四章 (4.2.2) 我们谈过自动取得 IP 的方式，设定真的很简单：

```
[root@clientlinux ~]# vim /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
NM_CONTROLLED=no
ONBOOT=yes
BOOTPROTO=dhcp  &lt;==就是他！指定这一个就对了！

[root@clientlinux ~]# /etc/init.d/network restart
```

同时记得要拿掉预设路由的设定喔！改完之后，就将我们的整个网络重新启动即可 (不要使用 ifdown 与 ifup ，因为还有预设路由要设定！)。请注意，如果你是在远程进行这个动作， 你的联机『肯定会挂掉！』，因为网络卡被你关了嘛！呵呵！所以请在本机前面才进行喔！如果执行的结果有找到正确的 DHCP 主机，那么几个档案可能会被更动喔：

```
# 1\. DNS 的 IP 会被更动呢！查阅一下 resolv.conf 先：
[root@clientlinux ~]# cat /etc/resolv.conf
search centos.vbird      &lt;==还记得设定过 domain-name 否？
domain centos.vbird      &lt;==还记得设定过 domain-name 否？
nameserver 168.95.1.1    &lt;==这就是我们在 dhcpd.conf内的设定值
nameserver 139.175.10.20

# 2\. 观察一下路由啦！
[root@clientlinux ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref   Use Iface
192.168.100.0   0.0.0.0         255.255.255.0   U     0      0       0 eth0
0.0.0.0         192.168.100.254 0.0.0.0         UG    0      0       0 eth0
# 嗯！没错！路由也被正确的捉到了！OK的啦！

# 3\. 察看一下客户端的指令吧！
[root@clientlinux ~]# netstat -tlunp &#124; grep dhc
Proto Recv-Q Send-Q Local Address   Foreign Address State  PID/Program name
udp        0      0 0.0.0.0:68      0.0.0.0:*              1694/dhclient
# 你没看错！确实是有个小程序在监测 DHCP 的联机状态呐！

# 4\. 看一看客户端租约所记载的信息吧！
[root@clientlinux ~]# cat /var/lib/dhclient/dhclient*
lease {
  interface "eth0";
  fixed-address 192.168.100.101; &lt;==取得的 IP 呦！
  option subnet-mask 255.255.255.0;
  option routers 192.168.100.254;
  option dhcp-lease-time 259200;
  option dhcp-message-type 5;
  option domain-name-servers 168.95.1.1,139.175.10.20;
  option dhcp-server-identifier 192.168.100.254;
  option domain-name "centos.vbird";
  renew 4 2011/07/28 05:01:24; &lt;==下一次预计更新 (renew) 的时间点
  rebind 5 2011/07/29 09:06:36;
  expire 5 2011/07/29 18:06:36;
}
# 这个档案会记录该适配卡所曾经要求过的 DHCP 信息喔！重要！
# 有没有看出来，他几乎就与你设定的 /etc/dhcp/dhcpd.conf 类似？ ^_^
```

有没有发现其实你的客户端取得的数据都被记载在 /var/lib/dhclient/dhclient*-eth0.leases 里头啊？ 如果你有多张网卡，那么每张网卡自己的 DHCP 要求就会被写入到不同档名的档案当中去！ 观察该档案就知道你的数据是如何啰！这可也是挺重要的呦！

Tips: 你或许会问说， dhcp 不是都会随机取得 IP 吗？那为什么这部客户端 clientlinux.centos.vbird 每次都能够取得相同的固定 IP 呢？ 很简单，因为上头的 dhclient-eth0.leases 里面的 fixed-address 指定了想要固定 IP 的选项。如果 DHCP 服务器的该 IP 没有被用走，也在规定的 range 设定值内，那就会发放给你这个 IP 了。如果你想要不同的 IP 呢？ 那就将你想要的 IP 取代上述的设定值啦！



例题：在文献中谈到，如果区网内有多个 DHCP 服务器 (假设有 DHCP1, DHCP2)，那么每次客户端对整个物理网络区段广播时，DHCP 服务器将是先抢先赢的局面。 但是若第一次取得 DHCP1 服务器的 IP 后，未来重新启动网络，都只会取得 DHCP1 的网络参数，这是为什

么？答：看到上述的 dhclient-eth0.leases 客户端档案了吗？因为你的主机想要取得上次取得的网络参数，因此将会对 DHCP1 要求网络参数。 如果你想要使用先抢先赢的方式来取得 IP ，或者想要使用 DHCP2 来取得 IP ，那么得要修订或者删除 dhclient-eth0.leases 才行。

12.3.2 客户端是 Windows

在 Windows 底下设定 DHCP 协议以取得 IP 实在是很简单喔！例如，你可以到第三章的 3.2.2 小节去瞧瞧如何设定的撷取图示。 我们这里以 Windows 7 作为介绍好了。你可以依据『开始』-->『控制台』-->『检视网络状态及工作』 -->『变更适配卡设定』 ，在出现的图示中，选择属于你的相关网卡，然后连击两下之后，就开始底下的设定程序：

1. 如上所述，点击网络卡设定后，会出现如下图示：



图 12.3-1、局域网络的 Windows 7 系统设定 DHCP 的方式

1. 在图 12.3-1 的地方按下箭头所指的『内容』处，就会出现如下画面啰：

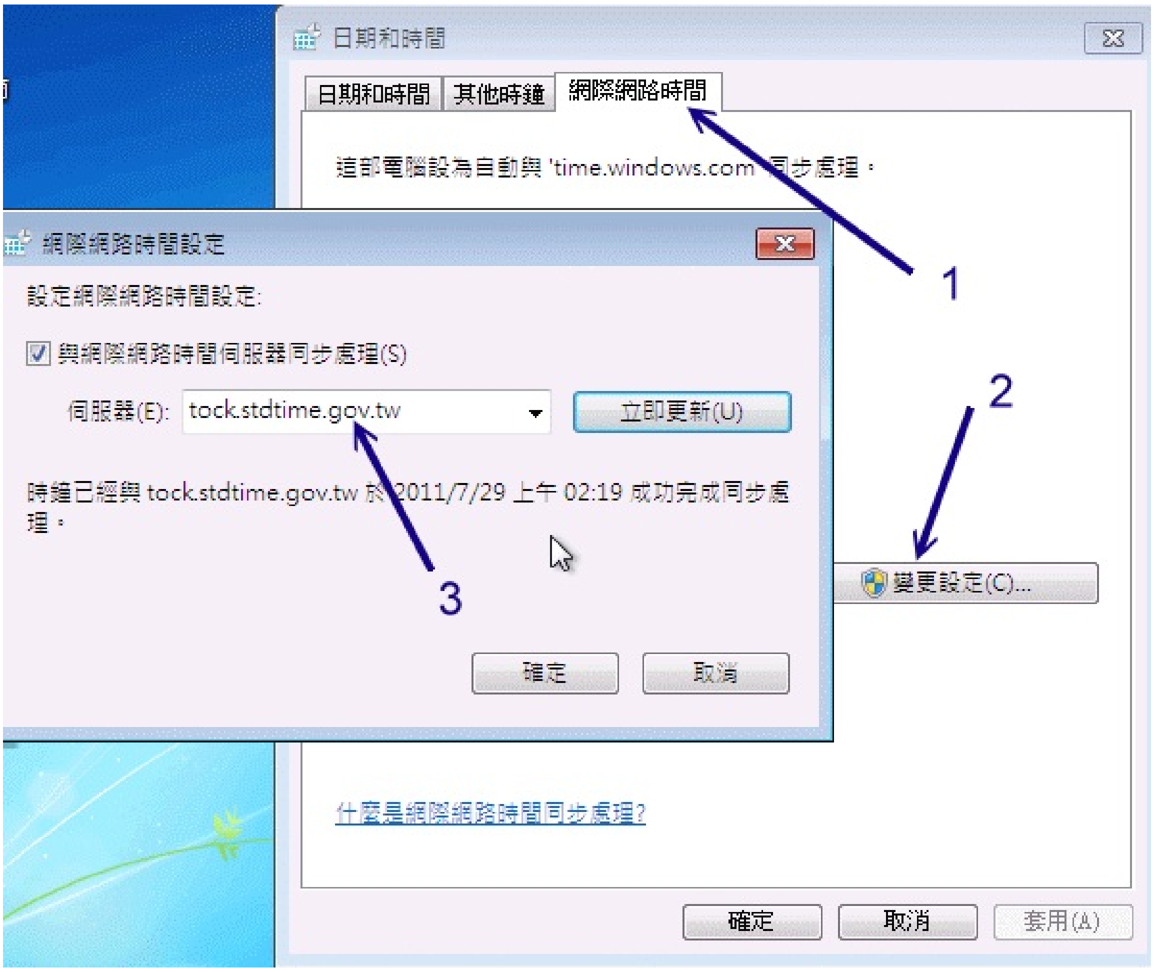


图 12.3-2、局域网络的 Windows 7 系统设定 DHCP 的方式

```
在上面的画面当中，先点选 TCP/IP4 第四版 IP 协议，然后按下『内容』就可以开始来修改网络参数啰！
```

1. 接下来如下图所示，你只要勾选『自动取得 IP 地址』那个项目，然后按下『确定』并离开设定画面， 如此一来 Windows 就会开始自动取得 IP 的工作了。

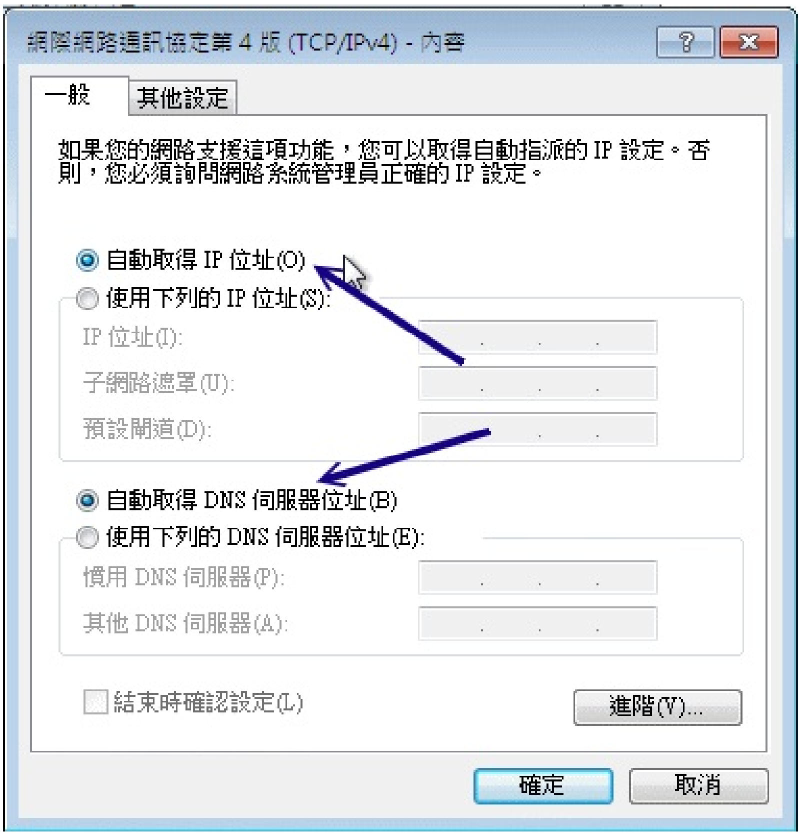


图 12.3-3、局域网络的 Windows 7 系统设定 DHCP 的方式

1. 那你如何确认你的 IP 已经被顺利的取得呢？如果是在早期的 Windows 95 ，你可以使用一个名为『 winipcfg 』 来观察你的 IP 设定。不过在 windows 2000 以后，你可能需要使用命令提示字符来观察才行。你可以使用： 『开始』--> 『所有程序』--> 『附属应用程序』--> 『命令提示字符』来取出终端机，然后这样处理看看：

```
C:\Users\win7&gt; ipconfig /all
....(前面省略)....
以太网络卡 区域联机:

   联机特定 DNS 后缀 . . . . . . . . . : centos.vbird
   描述 . . . . . . . . . . . . . . .: Intel(R) PRO/1000 MT Desktop Adapter
   实体地址 . . . . . . . . . . . . . .: 08-00-27-11-EB-C2
   DHCP 已启用 . . . . . . . . . . . : 是
   自动设定启用 . . . . . . . . . . . .: 是
   链接-本机 IPv6 地址 . . . . . . . . : fe80::ec92:b907:bc2a:a5fa%11(偏好选项)
   IPv4 地址 . . . . . . . . . . . . . : 192.168.100.30(偏好选项) &lt;==这是取得的IP
   子网掩码 . . . . . . . . . . . . .: 255.255.255.0
   租用取得 . . . . . . . . . . . . . .: 2011年7月27日 上午 11:59:18 &lt;==这是租约
   租用到期 . . . . . . . . . . . . . .: 2011年7月30日 上午 11:59:18
   预设网关 . . . . . . . . . . . . . .: 192.168.100.254
   DHCP 服务器 . . . . . . . . . . . . : 192.168.100.254  &lt;==这一部 DHCP 服务器
   DNS 服务器 . . . . . . . . . . . . .: 168.95.1.1       &lt;==取得的 DNS
                                          139.175.10.20
   NetBIOS over Tcpip . . . . . . . . .: 启用

C:\Users\win7&gt; ipconfig /renew
# 这样可以立即要求更新 IP 信息喔！
```

这样就 OK 的啦！简单吧！

12.4 DHCP 服务器端进阶观察与使用

如果你要管理的是几十部甚至是几百部的计算机时，你总是希望能够根据座位来进行 IP 的给予吧？因此，固定 IP 配合 MAC 就显的很重要啦！那么如何取得每部主机的 IP 呢？还有，你怎么查询到相关的租约呢？以及，如果你还想要进行远程开机， 帮使用者在固定的时间就开机呢？那就来看看底下的其他用途吧！

12.4.1 检查租约档案

客户端会主动的纪录租约信息，那服务器端更不能忘记记录啰！服务器端是记录在这个地方：

```
[root@www ~]# cat /var/lib/dhcpd/dhcpd.leases
lease 192.168.100.101 {
  starts 2 2011/07/26 18:06:36;  &lt;==租约开始日期
  ends 5 2011/07/29 18:06:36;    &lt;==租约结束日期
  tstp 5 2011/07/29 18:06:36;
  cltt 2 2011/07/26 18:06:36;
  binding state active;
  next binding state free;
  hardware ethernet 08:00:27:34:4e:44; &lt;==客户端网卡
}
```

从这个档案里面我们就知道有多少客户端已经向我们申请了 DHCP 的 IP 使用了呢！很容易了解吧！

12.4.2 让大量 PC 都具有固定 IP 的脚本

想一想，如果你有一百台计算机要管理，每部计算机都希望是固定 IP 的情况下，那你要如何处置？ 很简单，透过 DHCP 的 fixed-address 就行啦！但是，这一百台计算机的 MAC 如何取得？你要怎么改啦？ 难道每部计算机都去抄写，然后再回来设定 dhcpd.conf 吗？这也太可怕了吧？既然每部计算机最终都得要开机， 那么你在开机之后，利用手动的方法来设定好每部主机的 IP 后，在根据底下的脚本来处理好你的 dhcpd.conf 啰！

```
[root@www ~]# vim setup_dhcpd.conf
#!/bin/bash
read -p "Do you finished the IP's settings in every client (y/n)? " yn
read -p "How many PC's in this class (ex&gt; 60)? " num
if [ "$yn" = "y" ]; then
        for site in $(seq 1 ${num})
        do
                siteip="192.168.100.${site}"
                allip="$allip $siteip"
                ping -c 1 -w 1 $siteip &gt; /dev/null 2&gt;&1
                if [ "$?" == "0" ]; then
                        okip="$okip $siteip"
                else
                        errorip="$errorip $siteip"
                        echo "$siteip is DOWN"
                fi
        done
        [ -f dhcpd.conf ] && rm dhcpd.conf
        for site in $allip
        do
                pcname=pc$(echo $site &#124; cut -d '.' -f 4)
                mac=$(arp -n &#124; grep "$site " &#124; awk '{print $3}')
                echo "  host $pcname {"
                echo "          hardware ethernet ${mac};"
                echo "          fixed-address     ${site};"
                echo "  }"
                echo "  host $pcname {"                           &gt;&gt; dhcpd.conf
                echo "          hardware ethernet ${mac};"        &gt;&gt; dhcpd.conf
                echo "          fixed-address     ${site};"       &gt;&gt; dhcpd.conf
                echo "  }"                                        &gt;&gt; dhcpd.conf
        done
fi
echo "You can use dhcpd.conf (this directory) to modified your /etc/dhcp/dhcpd.conf"
echo "Finished."
```

这个脚本的想法很简单，如果你管理的计算机都是 Linux 的话，那么先开机后使用『 ifconfig eth0 YOURIP 』 来设定对应的 IP ，在鸟哥这个例子中，我使用的是 192.168.100.X/24 这个区段，此时 IP 就设定好了！ 然后在透过上面的脚本跑一次，每部计算机的 MAC 与 IP 对应就顺利的写入 dhcpd.conf 啰！ 然后你在将它贴上 /etc/dhcp/dhcpd.conf 即可！如果你管理的计算机是 Windows 的话， 那使用文字接口下达『 netsh interface ip set address xxx 』之类的指令来修订啰！

12.4.3 使用 ether-wake 实行远程自动开机 (remote boot)

既然已经知道客户端的 MAC 地址了，如果客户端的主机符合一些电源标准， 并且该客户端主机所使用之网络卡暨主板支持网络唤醒的功能时，我们就可以透过网络来让客户端计算机开机了。 如果你有一部主机想要让他可以透过网络来启动时，你必须要在这部客户端计算机上进行：

1. 首先你得要在 BIOS 里面设定『网络唤醒』的功能，否则是没有用的喔！
2. 再来你必须要让这部主机接上网络线，并且电源也是接通的。
3. 将这部主机的 MAC 抄下来，然后关机等待网络唤醒。

接下来请到永远开着的主机 DHCP 服务器上面 (其实只要任何一部 Linux 主机均可！) ，安装 net-tools 这个软件后， 就会取得 ether-wake 这个指令，这就是网络唤醒的主要功能！那该如何使用这个指令呢？假设客户端主机的 MAC 为 11:22:33:44:55:66 并且与我的服务器 eth1 相连接好了，那么你想要让这部主机被唤醒，就这样做吧：

```
[root@www ~]# ether-wake -i eth1 11:22:33:44:55:66

# 更多功能可以这样查阅喔：
[root@www ~]# ether-wake -u
```

然后你就会发现，哈哈！那部客户端主机被启动了！以后如果你要连到局域网络内的话， 只要能够连上你的防火墙主机，然后透过这个 ether-wake 软件，就能够让你局域网络内的主机启动了， 控管上面就更加方便的啦！你说是吧！ ^_^

Tips: 鸟哥办公室有一部桌机是经常用来测试的机器，但是因为比较耗电，因此当鸟哥离开办公室时，就会将计算机关闭。 不过鸟哥办公室有一部 NAT server 在负责防火墙的第一道关卡，当鸟哥在家里有需要查询到学校桌机的数据时， 桌机关了怎办？没关系，透过 NAT server 登入后，使用 ether-wake 唤醒桌机，那就能够开机进去工作啰！ 这样也比较不怕耗电问题～



12.4.4 DHCP 与 DNS 的关系

我们知道局域网络内如果很多 Linux 服务器时，你得要将 private IP 加入到每部主机的 /etc/hosts 里面， 这样在联机阶段的等待时间才不会有逾时或者是等待太久的问题。问题是，如果计算机数量太大，又有很多测试机时， 这时你得要常常去更新维护那些重灌过的机器的 /etc/hosts ，烦不烦呐？

此时在区网内架设一部 DNS 服务器负责主机名解析就很重要！因此既然已经有 DNS 服务器帮忙进行主机名的解析，那你根本不需要更动 /etc/hosts ！未来的新机器或者是新灌的计算机也不需要改写任何网络参数，这样维护会轻松很多。 因此，一个好的区网内，理论上，我们应该在 DHCP 服务器主机上面在安装一个 DNS 服务器，提供内部计算机的名称解析为宜。相关的设定就请参考第十九章 DNS 的介绍啰。

- DHCP 响应速度与有网管 switch 的设定问题

鸟哥在昆山信息传播系 (http://www.dic.ksu.edu.tw) 负责五间计算机教室的维护，每间计算机教室内部的 giga switch 是低阶的有网管功能的机器！有网管功能机器的设定信息比较多，switch 也能够进行封包异常的侦测与抵挡。问题是，如果抵挡的行为『太超过』时，也可能

造成许多问题。

鸟哥管理的计算机教室在重新启动网络取得 DHCP 时，都会等待几乎达 30 秒，虽然最终是成功的，但是等这么久呢！ 取得 IP 之后，网络速度却又是正常的，一切没问题～就是教导网络参数设定时，学生都会哇哇叫！以为失败了， 有的等了将近一分钟才告知取得 IP 且为正常...

后来问了有经验的计中的罗组长，才发现可能是 switch 的问题。大多在设定位于『L2 Features』-->『Spanning Tree』-->『STP Port Settings』的子项目之类的字眼，将 STP 之类的埠口都设定为关闭 (Disabled) 看看， 鸟哥做完这个设定后，DHCP 的取得就顺畅了！连带的网络开机功能也就没有问题～这部份也提供给大家参考呦！

Tips: 网友巩立伟兄来信谈到，STP 主要的目的是在抵挡广播风暴，若侦测到广播风暴时，该 switch 的埠口会被停用。 只是启动这个功能后，会较缓慢的进入运作状态，所以会产生较慢的情况发生。较好的 switch 会支援 RSTP (Rapid spanning tree protocol)，速度会较快一些。感谢朋友提供的信息喔！^_^



12.5 重点回顾

- DHCP (Dynamic Host Configuration Protocol) 可以提供网络参数给客户端计算机，使其自动设定网络的功能；
- 透过 DHCP 的统一管理，在同一网域当中就比较不容易出现 IP 冲突的情况发生；
- DHCP 可以透过 MAC 的比对来提供 Static IP (或称为固定 IP)，否则通常提供客户端 dynamic IP (或称为动态 IP)；
- DHCP 除了 Static IP 与 Dynamic IP 之外，还可以提供租约行为之设定；
- 在租约期限到期之前，客户端 dhcp 软件即会主动的要求更新 (约 0.5, 0.85 倍租约时间左右)；
- DHCP 可以提供的 MAC 比对、Dynamic IP 的 IP 范围以及租约期限等等，都在 dhcpd.conf 这个档案当中设定的；
- 一般的情况之下，使用者需要自行设定 dhcpd.leases 这个档案，不过，眞正的租约档案记录是在 /var/lib/dhclient/dhclient-eth0.leases 里面；
- 如果只是要单纯的 DHCP 服务，建议可以购买类似 IP 分享器的设备即可提供稳定且低耗电的网络服务。
- DHCP 服务与 DNS 服务的相关性很高；
- 若 DHCP 客户端取得 IP 的速度太慢，或许可以找一下有网管 switch 的 STP 设定值。

12.6 本章习题

- DHCP 主要的两种 IP 分配模式为何？主要的两种分配模式分别为 Dynamic IP 与 Static IP ， Static IP 透过 MAC 的比对，至于 Dynamic IP 则是直接取用网域中尚未被使用到的 IP 来进行 Client 端的分配。
- 在有 DHCP 主机存在的网域当中，且 client 端亦使用 DHCP 来规划客户端的网络参数，那么请问，在该网域当中，Client 端透过 DHCP 取得 IP 的流程为何？

  1. 首先， Client 端会发出一个 DHCP 要求封包；
  2. server 端接收到要求后，会主动的响应信息给 Client ；
  3. Client 若接受该 DHCP 主机所提供的参数，则主机会记录下租约信息，至于 client 端则开始以主机提供的参数设定其网络
- DHCP 是如何发送 Static IP 的？可以使用何种指令取得该信息？DHCP 主要利用网络卡的硬件地址，亦即俗称的『网络卡卡号』，也就是 MAC 来进行 Client 端的比对的，至于主动取得 Client 端的方式，可以透过 ping 以及 arp 来获得。

- 在 DHCP 服务器的租约档，亦即 /var/lib/dhcpd/dhcpd.leases 当中，记录了什么信息？这个档案主要记录了 Client 端连上 Server 端的纪录数据，他会被 DHCP 主机用来判定与 Client 端的租约行为喔！
- DHCP 的登录档放置于何处？就是最重要的 /var/log/messages 这个档案啦(预设状况下！)

12.7 参考数据与延伸阅读

- 注1：在维基百科的 DHCP 相关说明： http://zh.wikipedia.org/zh-tw/DHCP http://en.wikipedia.org/wiki/Dynamic_Host_Configuration_Protocol
- 其他可供查阅之数据： DHCP mini HOWTO：英文版：http://tldp.org/HOWTO/DHCP/index.html DHCP mini HOWTO：中文版：http://www.linux.org.tw/CLDP/MiniHOWTO/network/DHCP/ Internet Software consortium：http://www.isc.org/software/dhcp Study Area：http://www.study-area.org/linux/servers/linux_dhcp.htm Study Area 网络开机(我本善良兄撰写)：http://www.study-area.org/tips/wol.htm

---

2002/11/23：第一次完成 2003/03/15：加入相关重点回顾、与练习题 2003/09/10：修改版面去！ 2006/12/05：将旧的文章移动到此处 2006/12/06：好累！怎么老是在说累～总之，完成啰～新增加 ether-wake 的网络唤醒功能。 2011/02/24：将基于 CentOS 4.x 的旧的文章移动到此处 2011/03/03：加入了 DNS 的相关性，有网管的 STP 会影响 DHCP 的取得等数据。 2011/07/27：将基于 CentOS 5.x 的版本移动到此处 2011/07/27：主要是将 Windows 的图示以 Windows 7 来示意啰！

第十三章、文件服务器之一：NFS 服务器

最近更新日期：2011/07/27

NFS为 Network FileSystem 的简称，它的目的就是想让不同的机器、不同的操作系统可以彼此分享个别的档案啦！目前在 Unix Like 当中用来做为文件服务器是相当不错的一个方案喔！基本上， Unix Like 主机连接到另一部 Unix Like 主机来分享彼此的档案时，使用 NFS 要比 SAMBA 这个服务器快速且方便的多了！此外， NFS 的设定真的很简单，几乎只要记得启动 Remote Procedure Call 这个咚咚 (RPC, 就是 rpcbind 这个软件啦！) 就一定可以架设的起来！真是不错啊！ 如果是在 Linux PC cluster 的环境下，这个服务器被使用的机率更是高的多喔！所以得来玩一玩啊！

13.1 NFS 的由来与其功能

NFS 这个藉由网络分享文件系统的服务在架设的时候是很简单的，不过，它最大的问题在于『权限』方面的概念！ 因为在客户端与服务器端可能必须要具备相同的账号才能够存取某些目录或档案。 另外，NFS 的启动需要透过所谓的远程过程调用 (RPC)，也就是说，我们并不是只要启动 NFS 就好了， 还需要启动 RPC 这个服务才行啊！

因此，在开始进行 NFS 的设定之前，我们得先来了解一下，什么是 NFS 呢？不然讲了一堆也没有用，对吧！ ^_^！ 底下就来谈一谈什么是 NFS ，且 NFS 的启动还需要什么样的协议啊！

13.1.1 什么是 NFS (Network FileSystem)

NFS 就是 **N**etwork **F**ile**S**ystem 的缩写，最早之前是由 Sun 这家公司所发展出来的 (注1)。 它最大的功能就是可以透过网络，让不同的机器、不同的操作系统、可以彼此分享个别的档案 (share files)。所以，你也可以简单的将他看做是一个文件服务器 (file server) 呢！这个 NFS 服务器可以让你的 PC 来将网络远程的 NFS 服务器分享的目录，挂载到本地端的机器当中，在本地端的机器看起来，那个远程主机的目录就好像是自己的一个磁盘分区槽一样 (partition)！使用上面相当的便利！

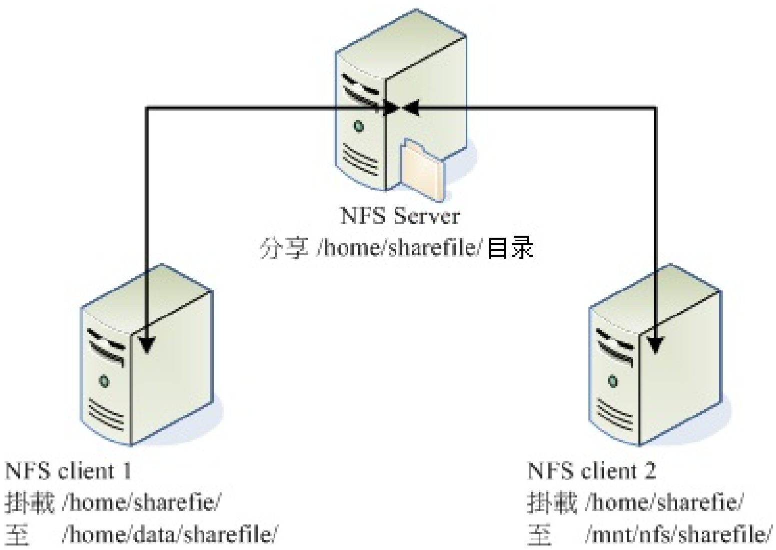


图 13.1-1、NFS 服务器分享目录与 Client 挂载示意图

就如同上面的图示一般，当我们的 NFS 服务器设定好了分享出来的 /home/sharefile 这个目录后，其他的 NFS 客户端就可以将这个目录挂载到自己系统上面的某个挂载点 (挂载点可以自定义)，例如前面图示中的 NFS client 1 与 NFS client 2 挂载的目录就不相同。我只要在 NFS client 1 系统中进入 /home/data/sharefile 内，就可以看到 NFS 服务器系统内的

/home/sharefile 目录下的所有数据了 (当然，权限要足够啊！^_^)！这个 /home/data/sharefile 就好像 NFS client 1 自己机器里面的一个 partition 喔！只要权限对了，那么你可以使用 cp, cd, mv, rm... 等等磁盘或档案相关的指令！眞是他 X 的方便呐！

好的，既然 NFS 是透过网络来进行数据的传输，那么经由第二章谈到的 socket pair 的概念你会知道 NFS 应该会使用一些埠口吧？那么 NFS 使用哪个埠口来进行传输呢？基本上 NFS 这个服务的埠口开在 2049 ，但是由于文件系统非常复杂，因此 NFS 还有其他的程序去启动额外的端口，但这些额外的端口启动的号码是？ 答案是....不知道！ @_@ ！因为预设 NFS 用来传输的埠口是随机选择小于 1024 以下的埠口来使用的。咦！那客户端怎么知道你服务器端使用那个埠口啊？此时就得要 远程过程调用 (Remote Procedure Call, RPC) 的协定来辅助啦！底下我们就来谈谈什么是 RPC？

13.1.2 什么是 RPC (Remote Procedure Call)

因为 NFS 支持的功能相当的多，而不同的功能都会使用不同的程序来启动， 每启动一个功能就会启用一些端口来传输数据，因此， NFS 的功能所对应的端口才没有固定住， 而是随机取用一些未被使用的小于 1024 的埠口来作为传输之用。但如此一来又造成客户端想要连上服务器时的困扰， 因为客户端得要知道服务器端的相关埠口才能够联机吧！

此时我们就得需要远程过程调用 (RPC) 的服务啦！RPC 最主要的功能就是在指定每个 NFS 功能所对应的 port number ，并且回报给客户端，让客户端可以连结到正确的埠口上去。 那 RPC 又是如何知道每个 NFS 的埠口呢？这是因为当服务器在启动 NFS 时会随机取用数个埠口，并主动的向 RPC 注册，因此 RPC 可以知道每个埠口对应的 NFS 功能，然后 RPC 又是固定使用 port 111 来监听客户端的需求并回报客户端正确的埠口， 所以当然可以让 NFS 的启动更为轻松愉快了！

Tips: 所以你要注意，要启动 NFS 之前，RPC 就要先启动了，否则 NFS 会无法向 RPC 注册。 另外，RPC 若重新启动时，原本注册的数据会不见，因此 RPC 重新启动后，它管理的所有服务都需要重新启动来重新向 RPC 注册。



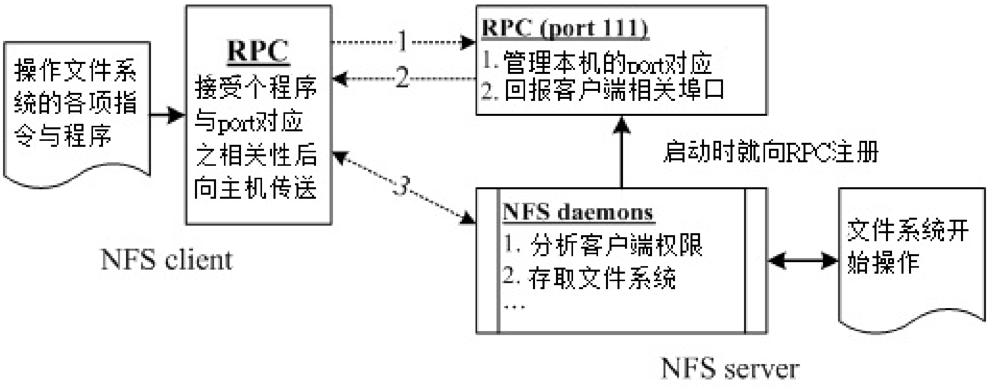


图 13.1-2、NFS 与 RPC 服务及文件系统操作的相关性

如上图所示，当客户端有 NFS 档案存取需求时，他会如何向服务器端要求数据呢？

1. 客户端会向服务器端的 RPC (port 111) 发出 NFS 档案存取功能的询问要求；
2. 服务器端找到对应的已注册的 NFS daemon 埠口后，会回报给客户端；
3. 客户端了解正确的埠口后，就可以直接与 NFS daemon 来联机。

由于 NFS 的各项功能都必须要向 RPC 来注册，如此一来 RPC 才能了解 NFS 这个服务的各项功能之 port number, PID, NFS 在服务器所监听的 IP 等等，而客户端才能够透过 RPC 的询问找到正确对应的埠口。 也就是说，NFS 必须要有 RPC 存在时才能成功的提供服务，因此我们称 NFS 为 RPC server 的一种。事实上，有很多这样的服务器都是向 RPC 注册的，举例来说，NIS (Network Information Service) 也是 RPC server 的一种呢。此外，由图 13.1-2 你也会知道，不论是客户端还是服务器端，要使用 NFS 时，两者都需要启动 RPC 才行喔！

更多的 NFS 相关协议信息你可以参考底下网页：

- RFC 1094, NFS 协议解释 http://www.faqs.org/rfcs/rfc1094.html
- Linux NFS-HOWTO：http://www.tldp.org/HOWTO/NFS-HOWTO/index.html

---

13.1.3 NFS 启动的 RPC daemons

我们现在知道 NFS 服务器在启动的时候就得要向 RPC 注册，所以 NFS 服务器也被称为 RPC server 之一。 那么 NFS 服务器主要的任务是进行文件系统的分享，文件系统的分享则与权限有关。 所以 NFS 服务器启动时至少需要两个 daemons ，一个管理客户端是否能够登入的问题， 一个管理客户端能够取得的权限。如果你还想要管理 quota 的话，那么 NFS 还得要再加载其他的 RPC 程序就是了。我们以较单纯的 NFS 服务器来说：

- rpc.nfsd： 最主要的 NFS 服务器服务提供商。这个 daemon 主要的功能就是在管理客户端是否能够使用服务器文件系统挂载信息等， 其中还包含这个登入者的 ID 的判别喔！

- rpc.mountd 这个 daemon 主要的功能，则是在管理 NFS 的文件系统哩！当客户端顺利的通过 rpc.nfsd 而登入服务器之后，在他可以使用 NFS 服务器提供的档案之前，还会经过档案权限 (就是那个 -rwxrwxrwx 与 owner, group 那几个权限啦) 的认证程序！他会去

读 NFS 的配置文件 /etc/exports 来比对客户端的权限，当通过这一关之后客户端就可以取得使用 NFS 档案的权限啦！(注：这个也是我们用来管理 NFS 分享之目录的权限与安全设定的地方哩！)

- rpc.lockd (非必要) 这个玩意儿可以用在管理档案的锁定 (lock) 用途。为何档案需要『锁定』呢？ 因为既然分享的 NFS 档案可以让客户端使用，那么当多个客户端同时尝试写入某个档案时， 就可能对于该档案造成一些问题啦！这个 rpc.lockd 则可以用来克服这个问题。 但 rpc.lockd 必须要同时在客户端与服务器端都开启才行喔！此外， rpc.lockd 也常与 rpc.statd 同时启用。
- rpc.statd (非必要) 可以用来检查档案的一致性，与 rpc.lockd 有关！若发生因为客户端同时使用同一档案造成档案可能有所损毁时， rpc.statd 可以用来检测并尝试回复该档案。与 rpc.lockd 同样的，这个功能必须要在服务器端与客户端都启动才会生效。

上述这几个 RPC 所需要的程序，其实都已经写入到两个基本的服务启动脚本中了，那就是 nfs 以及 nfslock 啰！ 亦即是在 /etc/init.d/nfs, /etc/init.d/nfslock，与服务器较有关的写入在 nfs 服务中，而与客户端的 rpc.lockd 之类的，就设定于 nfslock 服务中。

13.1.4 NFS 的档案访问权限

不知道你有没有想过这个问题，在图 13.1-1 的环境下，假如我在 NFS client 1 上面以 dmtsai 这个使用者身份想要去存取 /home/data/sharefile/ 这个来自 NFS server 所提供的文件系统时， 请问 NFS server 所提供的文件系统会让我以什么身份去存取？是 dmtsai 还是？

为什么会这么问呢？这是因为 NFS 本身的服务并没有进行身份登入的识别， 所以说，当你在客户端以 dmtsai 的身份想要存取服务器端的文件系统时， 服务器端会以客户端的使用者 UID 与 GID 等身份来尝试读取服务器端的文件系统。这时有个有趣的问题就产生啦！ 那就是如果客户端与服务器端的使用者身份并不一致怎么办？ 我们以底下这个图示来说明一下好了：

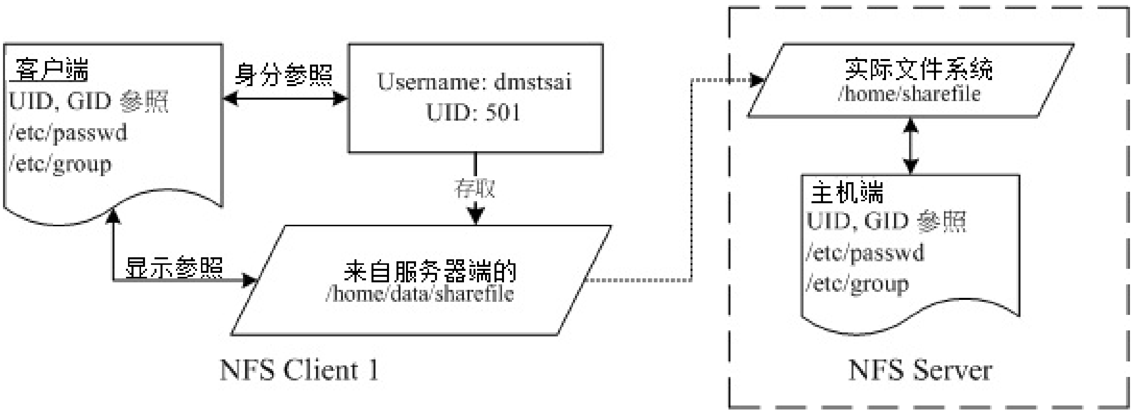


图 13.1-3、NFS 的服务器端与客户端的使用者身份确认机制

当我以 dmtsai 这个一般身份使用者要去存取来自服务器端的档案时，你要先注意到的是： 文件系统的 inode 所记录的属性为 UID, GID 而非账号与群组名。 那一般 Linux 主机会主动的以自己的 /etc/passwd, /etc/group 来查询对应的使用者、组名。 所以当 dmtsai 进入到该目录后，会参照 NFS client 1 的使用者与组名。 但是由于该目录的档案主要来自 NFS server ，所以可能就会发现几个情况：

- NFS server/NFS client 刚好有相同的账号与群组 则此时使用者可以直接以 dmtsai 的身份进行服务器所提供的文件系统之存取。

- NFS server 的 501 这个 UID 账号对应为 vbird 若 NFS 服务器上的 /etc/passwd 里面 UID 501 的使用者名称为 vbird 时， 则客户端的 dmtsai 可以存取服务器端的 vbird 这个使用者的档案喔！只因为两者具有相同的 UID 而已。这就造成很大的问题了！因为没有人可以保证客户端的 UID 所对应的账号会与服务器端相同， 那服务器所提供的数据不就可能会被错误的使用者乱改？

- NFS server 并没有 501 这个 UID 另一个极端的情况是，在服务器端并没有 501 这个 UID 的存在，则此时 dmtsai 的身份在该目录下会被压缩成匿名者， 一般 NFS 的匿名者会以 UID 为 65534 为其使用者，早期的 Linux distributions 这个 65534 的账号名称通常是 nobody ，我们的 CentOS 则取名为 nfsnobody 。但有时也会有特殊的情况，例如在服务器端分享 /tmp 的情况下， dmtsain 的身份还是会保持 501 但建立的各项数据在服务器端来看，就会属于无拥有者的资料。

- 如果使用者身份是 root 时 有个比较特殊的使用者，那就是每个 Linux 主机都有的 UID 为 0 的 root 。 想一想，如果客户端可以用 root 的身份去存取服务器端的文件系统时，那服务器端的数据哪有什么保护性？ 所以在预设的情况下， root 的身份会被主动的压缩成为匿名者。

总之，客户端使用者能做的事情是与 UID 及其 GID 有关的，那当客户端与服务器端的 UID 及账号的对应不一致时， 可能就会造成文件系统使用上的困扰，这个就是 NFS 文件系统在使用上面的一个很重要的地方！ 而在了解使用者账号与 UID 及文件系统的关系之后，要实际在客户端以 NFS 取用服务器端的文件系统时， 你还得需要具有：

- NFS 服务器有开放可写入的权限 (与 /etc/exports 设定有关)；
- 实际的档案权限具有可写入 (w) 的权限。

当你满足了 (1)使用者账号，亦即 UID 的相关身份； (2)NFS 服务器允许有写入的权限； (3) 文件系统确实具有 w 的权限时，你才具有该档案的可写入权限喔！ 尤其是身份 (UID) 确认的环节部分，最容易搞错啦！也因为如此， 所以 NFS 通常需要与 NIS (十四章) 这一个可以确认客户端与服务器端身份一致的服务搭配使用，以避免身份的错乱啊！ ^_^

Tips: 老实说，这个小节的数据比较难懂～尤其是刚刚接触到 NFS server 的朋友。因此，你可以先略过 13.1.4 这个小节。 但是，在你读完与做完本章后续所有的实作之后，记得回到这个小节来再查阅一次文章内容，相信会有进一步的认识的！


鸟哥
VBird

13.2 NFS Server 端的设定

既然要使用 NFS 的话，就得要安装 NFS 所需要的软件了！底下让我们查询一下系统有无安装所需要的软件， NFS 软件的架构以及如何设定 NFS 服务器吧！ ^_^

13.2.1 所需要的软件

以 CentOS 6.x 为例的话，要设定好 NFS 服务器我们必须要有两个软件才行，分别是：

- RPC 主程序：rpcbind

  就如同刚刚提的到，我们的 NFS 其实可以被视为一个 RPC 服务，而要启动任何一个 RPC 服务之前，我们都需要做好 port 的对应 (mapping) 的工作才行，这个工作其实就是『 rpcbind 』这个服务所负责的！也就是说， 在启动任何一个 RPC 服务之前，我们都需要启动 rpcbind 才行！ (在 CentOS 5.x 以前这个软件称为 portmap，在 CentOS 6.x 之后才称为 rpcbind 的！)

- NFS 主程序：nfs-utils

  就是提供 rpc.nfsd 及 rpc.mountd 这两个 NFS daemons 与其他相关 documents 与说明文件、执行文件等的软件！这个就是 NFS 服务所需要的主要软件啦！一定要有喔！

好了，知道我们需要这两个软件之后，现在干嘛？赶快去你的系统先用 RPM 看一下有没有这两个软件啦！ 没有的话赶快用 RPM 或 yum 去安装喔！不然就玩不下去了！

例题：请问我的主机是以 RPM 为套件管理的 Linux distribution ，例如 Red Hat, CentOS 与 SuSE 等版本，那么我要如何知道我的主机里面是否已经安装了 rpcbind 与 nfs 相关的软件呢？答：简单的使用『 rpm -qa | grep nfs 』与『 rpm -qa | grep rpcbind 』即可知道啦！如果没有安装的话， 在 CentOS 内可以使用『 yum install nfs-utils 』来安装！

13.2.2 NFS 的软件结构

NFS 这个咚咚真的是很简单，上面我们提到的 NFS 软件中，配置文件只有一个，执行档也不多， 记录文件也三三两两而已呐！赶紧先来看一看吧！ ^_^

- 主要配置文件：/etc/exports 这个档案就是 NFS 的主要配置文件了！不过，系统并没有默认值，所以这个档案『 不一定会存在』，你可能必须要使用 vim 主动的建立起这个档案喔！我们等一下要谈的设定也仅只是这个档案而已呐！

- NFS 文件系统维护指令：/usr/sbin/exportfs 这个是维护 NFS 分享资源的指令，我们可以利用这个指令重新分享 /etc/exports 变更的目录资源、将 NFS Server 分享的目录卸除或重新分享等等，这个指令是 NFS 系统里面相当重要的一个喔！至于指令的用法我们在底下会介绍。
- 分享资源的登录档：/var/lib/nfs/*tab 在 NFS 服务器的登录文件都放置到 /var/lib/nfs/ 目录里面，在该目录下有两个比较重要的登录档， 一个是 etab ，主要记录了 NFS 所分享出来的目录的完整权限设定值；另一个 xtab 则记录曾经链接到此 NFS 服务器的相关客户端数据。
- 客户端查询服务器分享资源的指令：/usr/sbin/showmount 这是另一个重要的 NFS 指令。exportfs 是用在 NFS Server 端，而 showmount 则主要用在 Client 端。这个 showmount 可以用来察看 NFS 分享出来的目录资源喔！

就说不难吧！主要就是这几个啰！

13.2.3 /etc/exports 配置文件的语法与参数

在开始 NFS 服务器的设定之前，你必须要了解的是，NFS 会直接使用到核心功能，所以你的核心必须要有支持 NFS 才行。万一如果你的核心版本小于 2.2 版，或者重新自行编译过核心的话，那么就得要很注意啦！因为你可能会忘记选择 NFS 的核心支持啊！

还好，我们 CentOS 或者是其他版本的 Linux ，预设核心通常是支持 NFS 功能的，所以你只要确认你的核心版本是目前新的 2.6.x 版，并且使用你的 distribution 所提供的核心，那应该就不会有问题啦！

Tips: 上面会提醒您这个问题的原因是，以前鸟哥都很喜欢自行编译一个特别的核心，但是某次编译核心时，却忘记加上了 NFS 的核心功能，结果 NFS server 无论如何也搞不起来～最后才想到原来俺的核心是非正规的...



至于 NFS 服务器的架设实在很简单，你只要编辑好主要配置文件 /etc/exports 之后，先启动 rpcbind (若已经启动了，就不要重新启动)，然后再启动 nfs ，你的 NFS 就成功了！ 不过这样的设定能否对客户端生效？那就得要考虑你权限方面的设定能力了。废话少说，我们就直接来看看那个 /etc/exports 应该如何设定吧！某些 distributions 并不会主动提供 /etc/exports 档案，所以请你自行手动建立它吧。

```
[root@www ~]# vim /etc/exports
/tmp         192.168.100.0/24(ro)   localhost(rw)   *.ev.ncku.edu.tw(ro,sync)
[分享目录]   [第一部主机(权限)]     [可用主机名]    [可用通配符]
```

你看看，这个配置文件有够简单吧！每一行最前面是要分享出来的目录，注意喔！是以目录为单位啊！ 然后这个目录可以依照不同的权限分享给不同的主机，像鸟哥上面的例子说明是： 要将 /tmp 分别分享给三个不同的主机或网域的意思。记得主机后面以小括号 () 设计权限参数， 若权限参数不止一个时，则以逗号 (,) 分开。且主机名与小括号是连在一起的喔！在这个档案内也可以利用 # 来批注呢。

至于主机名的设定主要有几个方式：

- 可以使用完整的 IP 或者是网域，例如 192.168.100.10 或 192.168.100.0/24 ，或 192.168.100.0/255.255.255.0 都可以接受！
- 也可以使用主机名，但这个主机名必须要在 /etc/hosts 内，或可使用 DNS 找到该名称才行啊！反正重点是可找到 IP 就是了。如果是主机名的话，那么他可以支持通配符，例如 * 或 ? 均可接受。

至于权限方面 (就是小括号内的参数) 常见的参数则有：

参数值	内容说明
rw ro	该目录分享的权限是可擦写 (read-write) 或只读 (read-only)，但最终能不能读写，还是与文件系统的 rwx 及身份有关。
sync async	sync 代表数据会同步写入到内存与硬盘中，async 则代表数据会先暂存于内存当中，而非直接写入硬盘！
no_root_squash root_squash	客户端使用 NFS 文件系统的账号若为 root 时，系统该如何判断这个账号的身份？预设的情况下，客户端 root 的身份会由 root_squash 的设定压缩成 nfsnobody， 如此对服务器的系统会较有保障。但如果你想要开放客户端使用 root 身份来操作服务器的文件系统，那么这里就得要开 no_root_squash 才行！
all_squash	不论登入 NFS 的使用者身份为何， 他的身份都会被压缩成为匿名用户，通常也就是 nobody(nfsnobody) 啦！
anonuid anongid	anon 意指 anonymous (匿名者) 前面关于 *_squash 提到的匿名用户的 UID 设定值，通常为 nobody(nfsnobody)，但是你可以自行设定这个 UID 的值！当然，这个 UID 必需要存在于你的 /etc/passwd 当中！ anonuid 指的是 UID 而 anongid 则是群组的 GID 啰。

这是几个比较常见的权限参数，如果你有兴趣玩其他的参数时，请自行 man exports 可以发现很多有趣的数据。 接下来我们利用上述的几个参数来实际思考一下几个有趣的小习题：

例题一：让 root 保有 root 的权限我想将 /tmp 分享出去给大家使用，由于这个目录本来就是大家都可以读写的，因此想让所有的人都可以存取。此外，我要让 root 写入的档案还是具有 root 的权限，那如何设计配置文件？答：

```
[root@www ~]# vim /etc/exports
# 任何人都可以用我的 /tmp ，用通配符来处理主机名，重点在 no_root_squash
/tmp  *(rw,no_root_squash)
```

主机名可以使用通配符，上头表示无论来自哪里都可以使用我的 /tmp 这个目录。 再次提醒，『 *(rw,no_root_squash)* 』这一串设定值中间是没有空格符的喔！而 */tmp* 与 (rw,no_root_squash) 则是有空格符来隔开的！特别注意到那个 no_root_squash 的功能！在这个例子中，如果你是客户端，而且你是以 root 的身份登入你的 Linux 主机，那么当你 mount 上我这部主机的 /tmp 之后，你在该 mount 的目录当中，将具有『root 的权限！』

例题二：同一目录针对不同范围开放不同权限我要将一个公共的目录 /home/public 公开出去，但是只有限定我的局域网络 192.168.100.0/24 这个网域且加入 vbirdgroup (第一章的例题建立的群组) 的用户才能够读写，其他来源则只能读取。答：

```
[root@www ~]# mkdir /home/public
[root@www ~]# setfacl -m g:vbirdgroup:rwx /home/public
[root@www ~]# vim /etc/exports
/tmp          *(rw,no_root_squash)
/home/public  192.168.100.0/24(rw)    *(ro)
# 继续累加在后面，注意，我有将主机与网域分为两段 (用空白隔开) 喔！
```

上面的例子说的是，当我的 IP 是在 192.168.100.0/24 这个网段的时候，那么当我在 Client 端挂载了 Server 端的 /home/public 后，针对这个被我挂载的目录我就具有可以读写的权限～ 至于如果我不是在这个网段之内，那么这个目录的数据我就仅能读取而已，亦即为只读的属性啦！

需要注意的是，通配符仅能用在主机名的分辨上面，IP 或网段就只能用 192.168.100.0/24 的状况， 不可以使用 192.168.100.* 喔！

例题三：仅给某个单一主机使用的目录设定我要将一个私人的目录 /home/test 开放给 192.168.100.10 这个 Client 端的机器来使用时，该如何设定？ 假设使用者的身份是 dmtsai 才具有完整的权限时。答：

```
[root@www ~]# mkdir /home/test
[root@www ~]# setfacl -m u:dmtsai:rwx /home/test
[root@www ~]# vim /etc/exports
/tmp          *(rw,no_root_squash)
/home/public  192.168.100.0/24(rw)    *(ro)
/home/test    192.168.100.10(rw)
# 只要设定 IP 正确即可！
```

这样就设定完成了！而且，只有 192.168.100.10 这部机器才能对 /home/test 这个目录进行存取喔！

例题四：开放匿名登录的情况我要让 *.centos.vbird 网域的主机，登入我的 NFS 主机时，可以存取 /home/linux ，但是他们存数据的时候，我希望他们的 UID 与 GID 都变成 45 这个身份的使用者，假设我 NFS 服务器上的 UID 45 与 GID 45 的用户/组名为 nfsanon。答：

```
[root@www ~]# groupadd -g 45 nfsanon
[root@www ~]# useradd -u 45 -g nfsanon nfsanon
[root@www ~]# mkdir /home/linux
[root@www ~]# setfacl -m u:nfsanon:rwx /home/linux
[root@www ~]# vim /etc/exports
/tmp          *(rw,no_root_squash)
/home/public  192.168.100.0/24(rw)    *(ro)
/home/test    192.168.100.10(rw)
/home/linux   *.centos.vbird(rw,all_squash,anonuid=45,anongid=45)
# 如果要开放匿名，那么重点是 all_squash，并且要配合 anonuid 喔！
```

特别注意到那个 all_squash 与 anonuid, anongid 的功能！如此一来，当 clientlinux.centos.vbird 登入这部 NFS 主机，并且在 /home/linux 写入档案时，该档案的所有人与所有群组，就会变成 /etc/passwd 里面对应的 UID 为 45 的那个身份的使用者了！

上面四个案例的权限如果依照13.1.4 存取设定权限来思考的话， 那么权限会是什么情况呢？让我们来检查一下：

- 客户端与服务器端具有相同的 UID 与账号：

假设我在 192.168.100.10 登入这部 NFS (IP 假设为 192.168.100.254) 服务器，并且我在 192.168.100.10 的账号为 dmtsai 这个身份，同时，在这部 NFS 上面也有 dmtsai 这个账号，并具有相同的 UID ，果真如此的话，那么：

1. 由于 192.168.100.254 这部 NFS 服务器的 /tmp 权限为 -rwxrwxrwt ，所以我 (dmtsai 在 192.168.100.10 上面) 在 /tmp 底下具有存取的权限，并且写入的档案所有人为 dmtsai ；
2. 在 /home/public 当中，由于我有读写的权限，所以如果在 /home/public 这个目录的权限对于 dmtsai 有开放写入的话，那么我就可以读写，并且我写入的档案所有人是 dmtsai 。但是万一 /home/public 对于 dmtsai 这个使用者并没有开放可以写入的权限时， 那么我还是没有办法写入档案喔！这点请特别留意！
3. 在 /home/test 当中，我的权限与 /home/public 相同的状态！还需要 NFS 服务器的 /home/test 对于 dmtsai 有开放权限；
4. 在 /home/linux 当中就比较麻烦！因为不论你是何种 user ，你的身份一定会被变成 UID=45 这个账号！所以，这个目录就必需要针对 UID = 45 的那个账号名称，修改他的权限才行！

5. 客户端与服务器端的账号并未相同时：

假如我在 192.168.100.10 的身份为 vbird (uid 为 600)，但是 192.168.100.254 这部 NFS 主机却没有 uid=600 的账号时，情况会变成怎样呢？

1. 我在 /tmp 底下还是可以写入，只是该档案的权限会保持为 UID=600 ，因此服务器端看起来就会怪怪的， 因为找不到 UID=600 这个账号的显示，故档案拥有者会填上 600 呦！
2. 我在 /home/public 里面是否可以写入，还需要视 /home/public 的权限而定，不过，由于没有加上 all_squash 的参数， 因此在该目录下会保留客户端的使用者 UID，同上一点所示。

3. /home/test 的观点与 /home/public 相同！
4. /home/linux 底下，我的身份就被变成 UID = 45 那个使用者就是了！

5. 当客户端的身份为 root 时：

假如我在 192.168.100.10 的身份为 root 呢？ root 这个账号每个系统都会有呀！权限变成怎样呢？

1. 我在 /tmp 里面可以写入，并且由于 no_root_squash 的参数，改变了预设的 root_squash 设定值，所以在 /tmp 写入的档案所有人为 root 喔！
2. 我在 /home/public 底下的身份还是被压缩成为 nobody 了！因为默认属性里面都具有 root_squash 呢！所以，如果 /home/public 有针对 nobody 开放写入权限时，那么我就可以写入，但是档案所有人变成 nobody 就是了！
3. /home/test 与 /home/public 相同；
4. /home/linux 的情况中，我 root 的身份也被压缩成为 UID = 45 的那个使用者了！

这样的权限讲解之后，你可以了解了吗？这里是最重要的地方，如果这一关通过了，底下的咚咚就没有问题啦！ ^_^！ 在你将本文读完后，最好还是回到13.1.4 NFS 的档案访问权限好好的瞧一瞧， 才能解决 NFS 的问题喔！

13.2.4 启动 NFS

配置文件搞定后，当然要开始来启动才行啊！而前面我们也提到过，NFS 的启动还需要 rpcbind 的协助才行啊！ 所以赶紧来启动吧！

```
[root@www ~]# /etc/init.d/rpcbind start
# 如果 rpcbind 本来就已经在执行了，那就不需要启动啊！

[root@www ~]# /etc/init.d/nfs start
# 有时候某些 distributions 可能会出现如下的警告讯息：
exportfs: /etc/exports [3]: No 'sync' or 'async' option specified
for export "192.168.100.10:/home/test".
  Assuming default behaviour ('sync').
# 上面的警告讯息仅是在告知因为我们没有指定 sync 或 async 的参数，
# 则 NFS 将默认会使用 sync 的信息而已。你可以不理他，也可以加入 /etc/exports。

[root@www ~]# /etc/init.d/nfslock start
[root@www ~]# chkconfig rpcbind on
[root@www ~]# chkconfig nfs on
[root@www ~]# chkconfig nfslock on
```

那个 rpcbind 根本就不需要设定！只要直接启动它就可以啦！启动之后，会出现一个 port 111 的 sunrpc 的服务，那就是 rpcbind 啦！至于 nfs 则会启动至少两个以上的 daemon 出现！然后就开始在监听 Client 端的需求啦！你必须要很注意屏幕上面的输出信息， 因为如果配置文件写错的话，屏幕上会显示出错误的地方喔！

此外，如果你想要增加一些 NFS 服务器的数据一致性功能时，可能需要用到 rpc.lockd 及 rpc.statd 等 RPC 服务， 那么或许你可以增加一个服务，那就是 nfslock 啰！启动之后，请赶快到 /var/log/messages 里面看看有没有被正确的启动呢？

```
[root@www ~]# tail /var/log/messages
Jul 27 17:10:39 www kernel: Installing knfsd (copyright (C) 1996 okir@monad.swb.de).
Jul 27 17:10:54 www kernel: NFSD: Using /var/lib/nfs/v4recovery as the NFSv4 state
recovery directory
Jul 27 17:10:54 www kernel: NFSD: starting 90-second grace period
Jul 27 17:11:32 www rpc.statd[3689]: Version 1.2.2 starting
```

在确认启动没有问题之后，接下来我们来瞧一瞧那么 NFS 到底开了哪些埠口？

```
[root@www ~]# netstat -tulnp&#124; grep -E '(rpc&#124;nfs)'
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address  Foreign Address  State   PID/Program name
tcp        0      0 0.0.0.0:875    0.0.0.0:*        LISTEN  3631/rpc.rquotad
tcp        0      0 0.0.0.0:111    0.0.0.0:*        LISTEN  3601/rpcbind
tcp        0      0 0.0.0.0:48470  0.0.0.0:*        LISTEN  3647/rpc.mountd
tcp        0      0 0.0.0.0:59967  0.0.0.0:*        LISTEN  3689/rpc.statd
tcp        0      0 0.0.0.0:2049   0.0.0.0:*        LISTEN  -
udp        0      0 0.0.0.0:875    0.0.0.0:*                3631/rpc.rquotad
udp        0      0 0.0.0.0:111    0.0.0.0:*                3601/rpcbind
udp        0      0 0.0.0.0:897    0.0.0.0:*                3689/rpc.statd
udp        0      0 0.0.0.0:46611  0.0.0.0:*                3647/rpc.mountd
udp        0      0 0.0.0.0:808    0.0.0.0:*                3601/rpcbind
udp        0      0 0.0.0.0:46011  0.0.0.0:*                3689/rpc.statd
```

注意看到上面喔！总共产生了好多的 port 喔！真是可怕！不过主要的埠口是：

- rpcbind 启动的 port 在 111 ，同时启动在 UDP 与 TCP；
- nfs 本身的服务启动在 port 2049 上头！
- 其他 rpc.* 服务启动的 port 则是随机产生的，因此需向 port 111 注册。

好了，那我怎么知道每个 RPC 服务的注册状况？没关系，你可以使用 rpcinfo 来观察的。

```
[root@www ~]# rpcinfo -p [IP&#124;hostname]
[root@www ~]# rpcinfo -t&#124;-u  IP&#124;hostname 程序名称
选项与参数：
-p ：针对某 IP (未写则预设为本机) 显示出所有的 port 与 porgram 的信息；
-t ：针对某主机的某支程序检查其 TCP 封包所在的软件版本；
-u ：针对某主机的某支程序检查其 UDP 封包所在的软件版本；

# 1\. 显示出目前这部主机的 RPC 状态
[root@www ~]# rpcinfo -p localhost
   program vers proto   port  service
    100000    4   tcp    111  portmapper
    100000    3   tcp    111  portmapper
    100000    2   tcp    111  portmapper
    100000    4   udp    111  portmapper
    100000    3   udp    111  portmapper
    100000    2   udp    111  portmapper
    100011    1   udp    875  rquotad
    100011    2   udp    875  rquotad
    100011    1   tcp    875  rquotad
    100011    2   tcp    875  rquotad
    100003    2   tcp   2049  nfs
....(底下省略)....
# 程序代号 NFS版本 封包类型 埠口  服务名称

# 2\. 针对 nfs 这个程序检查其相关的软件版本信息 (仅察看 TCP 封包)
[root@www ~]# rpcinfo -t localhost nfs
program 100003 version 2 ready and waiting
program 100003 version 3 ready and waiting
program 100003 version 4 ready and waiting
# 可发现提供 nfs 的版本共有三种，分别是 2, 3, 4 版呦！
```

仔细瞧瞧，上面出现的信息当中除了程序名称与埠口的对应可以与 netstat -tlunp 输出的结果作比对之外，还需要注意到 NFS 的版本支持！新的 NFS 版本传输速度较快，由上表看起来，我们的 NFS 至少支持到第 4 版，应该还算合理啦！ ^_^！ 如果你的 rpcinfo 无法输出，那就表示注册的数据有问题啦！可能需要重新启动 rpcbind 与 nfs 喔！

13.2.5 NFS 的联机观察

在你的 NFS 服务器设定妥当之后，我们可以在 server 端先自我测试一下是否可以联机喔！就是利用 showmount 这个指令来查阅！

```
[root@www ~]# showmount [-ae] [hostname&#124;IP]
选项与参数：
-a ：显示目前主机与客户端的 NFS 联机分享的状态；
-e ：显示某部主机的 /etc/exports 所分享的目录数据。

# 1\. 请显示出刚刚我们所设定好的相关 exports 分享目录信息
[root@www ~]# showmount -e localhost
Export list for localhost:
/tmp         *
/home/linux  *.centos.vbird
/home/test   192.168.100.10
/home/public (everyone)
```

很简单吧！所以，当你要扫瞄某一部主机他提供的 NFS 分享的目录时，就使用 showmount -e IP (或hostname) 即可！非常的方便吧！这也是 NFS client 端最常用的指令喔！ 另外，NFS 关于目录权限设定的数据非常之多！在 /etc/exports 只是比较特别的权限参数而已，还有很多预设参数呢！ 这些预设参数在哪？我们可以检查一下 /var/lib/nfs/etab 就知道了！

```
[root@www ~]# tail /var/lib/nfs/etab
/home/public    192.168.100.0/24(rw,sync,wdelay,hide,nocrossmnt,secure,root_squash,
no_all_squash,no_subtree_check,secure_locks,acl,anonuid=65534,anongid=65534)
# 上面是同一行，可以看出除了 rw, sync, root_squash 等等，
# 其实还有 anonuid 及 anongid 等等的设定！
```

上面仅仅是一个小范例，透过分析 anonuid=65534 对比 /etc/passwd 后，会发现 CentOS 出现的是 nfsnobody 啦！这个账号在不同的版本都可能会不一样的！另外，如果有其他客户端挂载了你的 NFS 文件系统时，那么该客户端与文件系统信息就会被记录到 /var/lib/nfs/xtab 里头去的！

另外，如果你想要重新处理 /etc/exports 档案，当重新设定完 /etc/exports 后需不需要重新启动 nfs ？ 不需要啦！如果重新启动 nfs 的话，要得再向 RPC 注册！很麻烦～这个时候我们可以透过 exportfs 这个指令来帮忙喔！

```
[root@www ~]# exportfs [-aruv]
选项与参数：
-a ：全部挂载(或卸除) /etc/exports 档案内的设定
-r ：重新挂载 /etc/exports 里面的设定，此外，亦同步更新 /etc/exports
     及 /var/lib/nfs/xtab 的内容！
-u ：卸除某一目录
-v ：在 export 的时候，将分享的目录显示到屏幕上！

# 1\. 重新挂载一次 /etc/exports 的设定
[root@www ~]# exportfs -arv
exporting 192.168.100.10:/home/test
exporting 192.168.100.0/24:/home/public
exporting *.centos.vbird:/home/linux
exporting *:/home/public
exporting *:/tmp

# 2\. 将已经分享的 NFS 目录资源，通通都卸除
[root@www ~]# exportfs -auv
# 这时如果你再使用 showmount -e localhost 就会看不到任何资源了！
```

要熟悉一下这个指令的用法喔！这样一来，就可以直接重新 exportfs 我们的记录在 /etc/exports 的目录数据啰！但是要特别留意，如果你仅有处理配置文件，但并没有相对应的目录 (/home/public 等目录) 可以提供使用啊！ 那可能会出现一些警告讯息喔！所以记得要建立分享的目录才对！

13.2.6 NFS 的安全性

在 NFS 的安全性上面，有些地方是你必须要知道的喔！底下我们分别来谈一谈：

- 防火墙的设定问题与解决方案：

一般来说， NFS 的服务仅会对内部网域开放，不会对因特网开放的。然而，如果你有特殊需求的话， 那么也可能会跨不同网域就是了。但是，NFS 的防火墙特别难搞，为什么呢？因为除了固定的 port 111, 2049 之外， 还有很多不固定的埠口是由 rpc.mountd, rpc.rquotad 等服务所开启的，所以，你的 iptables 就很难设定规则！ 那怎办？难道整个防火墙机制都要取消才可以？

为了解决这个问题， CentOS 6.x 有提供一个固定特定 NFS 服务的埠口配置文件，那就是 /etc/sysconfig/nfs 啦！ 你在这个档案里面就能够指定特定的埠口，这样每次启动 nfs 时，相关服务启动的埠口就会固定，如此一来， 我们就能够设定正确的防火墙啰！这个配置文件内容很多，绝大部分的数据你都不要去更改，只要改跟 PORT 这个关键词有关的数据即可。 那么需要更改的 rpc 服务有哪些呢？主要有 mountd, rquotad, nlockmgr 这三个，所以你应该要这样改：

```
[root@www ~]# vim /etc/sysconfig/nfs
RQUOTAD_PORT=1001   &lt;==约在 13 行左右
LOCKD_TCPPORT=30001 &lt;==约在 21 行左右
LOCKD_UDPPORT=30001 &lt;==约在 23 行左右
MOUNTD_PORT=1002    &lt;==约在 41 行左右
# 记得设定值最左边的批注服务要拿掉之外，埠口的值你也可以自行决定。

[root@www ~]# /etc/init.d/nfs restart
[root@www ~]# rpcinfo -p &#124; grep -E '(rquota&#124;mount&#124;nlock)'
    100011    2   udp   1001  rquotad
    100011    2   tcp   1001  rquotad
    100021    4   udp  30001  nlockmgr
    100021    4   tcp  30001  nlockmgr
    100005    3   udp   1002  mountd
    100005    3   tcp   1002  mountd
# 上述的输出数据已经被鸟哥汇整过了，没用到的埠口先挪掉了啦！
```

很可怕吧！如果想要开放 NFS 给别的网域的朋友使用，又不想要让对方拥有其他服务的登入功能， 那你的防火墙就得要开放上述的十个埠口啦！有够麻烦的～假设你想要开放 120.114.140.0/24 这个网域的人能够使用你这部服务器的 NFS 的资源，且假设你已经使用第九章提供的防火墙脚本， 那么你还得要这样做才能够针对该网域放行喔：

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.allow
iptables -A INPUT -i $EXTIF -p tcp -s 120.114.140.0/24 -m multiport \
         --dport 111,2049,1001,1002,30001 -j ACCEPT
iptables -A INPUT -i $EXTIF -p udp -s 120.114.140.0/24 -m multiport \
         --dport 111,2049,1001,1002,30001 -j ACCEPT

[root@www ~]# /usr/local/virus/iptables/iptables.rule
# 总是要重新执行这样防火墙规则才会顺利的生效啊！别忘记！别忘记！
```

- 使用 /etc/exports 设定更安全的权限：

这就牵涉到你的逻辑思考了！怎么设定都没有关系，但是在『便利』与『安全』之间，要找到你的平衡点呐！善用 root_squash 及 all_squash 等功能，再利用 anonuid 等等的设定来规范登入你主机的用户身份！应该还是有办法提供一个较为安全的 NFS 服务器的！

另外，当然啦，你的 NFS 服务器的文件系统之权限设定也需要很留意！ 不要随便设定成为 -rwxrwxrwx ，这样会造成你的系统『很大的困扰』的啊！

- 更安全的 partition 规划：

如果你的工作环境中，具有多部的 Linux 主机，并且预计彼此分享出目录时，那么在安装 Linux 的时候，最好就可以规划出一块 partition 作为预留之用。因为『 NFS 可以针对目录来分享』，因此，你可以将预留的 partition 挂载在任何一个挂载点，再将该挂载点 (就是目录啦！)由 /etc/exports 的设定中分享出去，那么整个工作环境中的其他 Linux 主机就可以使用该 NFS 服务器的那块预留的 partition 了！所以，在主机的规划上面，主要需要留意的只有 partition 而已。此外，由于分享的 partition 可能较容易被入侵，最好可以针对该 partition 设定比较严格的参数在 /etc/fstab 当中喔！

此外，如果你的分割做的不够好，举例来说，很多人都喜欢使用懒人分割法，亦即整个系统中只有一个根目录的 partition 而已。这样做会有什么问题呢？假设你分享的是 /home 这个给一般用户的目录好了，有些用户觉得这个 NFS 的磁盘太好用了， 结果使用者就将他的一大堆暂存数据通通塞进这个 NFS 磁盘中。想一想，如果整个根目录就因为这个 /home 被塞爆了，那么你的系统将会造成无法读写的困扰。因此，一个良好的分割规划，或者是利用磁盘配额来限制还是很重要的工作。

- NFS 服务器关机前的注意事项：

需要注意的是，由于 NFS 使用的这个 RPC 服务，当客户端连上服务器时，那么你的服务器想要关机， 那可就会成为『不可能的任务』！如果你的服务器上面还有客户端在联机，那么你要关机， 可能得要等到数个钟头才能够正常的关机成功！嗄！真的假的！不相信吗？不然你自个儿试试看！^_^！

所以啰，建议你的 NFS Server 想要关机之前，能先『关掉 rpcbind 与 nfs 』这两个东西！ 如果无法正确的将这两个 daemons 关掉，那么先以 netstat -utlp 找出 PID ，然后以 kill 将他关掉先！这样才有办法正常的关机成功喔！这个请特别特别的注意呢！

当然啦，你也可以利用 showmount -a localhost 来查出来那个客户端还在联机？ 或者是查阅 /var/lib/nfs/rmtab 或 xtab 等档案来检查亦可。找到这些客户端后， 可以直接 call 他们啊！让他们能够帮帮忙先！ ^_^

事实上，客户端以 NFS 联机到服务器端时，如果他们可以下达一些比较不那么『硬』的挂载参数时， 就能够减少这方面的问题喔！相关的安全性可以参考下一小节的 客户端可处理的挂载参数与开机挂载。

13.3 NFS 客户端的设定

既然 NFS 服务器最主要的工作就是分享文件系统给网络上其他的客户端，所以客户端当然得要挂载这个玩意儿啰！ 此外，服务器端可以加设防火墙来保护自己的文件系统，那么客户端挂载该文件系统后，难道不需要保护自己？ 呵呵！所以底下我们要来谈一谈几个 NFS 客户端的课题。

13.3.1 手动挂载 NFS 服务器分享的资源

你要如何挂载 NFS 服务器所提供的文件系统呢？基本上，可以这样做：

1. 确认本地端已经启动了 rpcbind 服务！
2. 扫瞄 NFS 服务器分享的目录有哪些，并了解我们是否可以使用 (showmount)；
3. 在本地端建立预计要挂载的挂载点目录 (mkdir)；
4. 利用 mount 将远程主机直接挂载到相关目录。

好，现在假设客户端在 192.168.100.10 这部机器上，而服务器是 192.168.100.254 ， 那么赶紧来检查一下我们是否已经有 rpcbind 的启动，另外远程主机有什么可用的目录呢！

```
# 1\. 启动必备的服务：若没有启动才启动，有启动则保持原样不动。
[root@clientlinux ~]# /etc/init.d/rpcbind start
[root@clientlinux ~]# /etc/init.d/nfslock start
# 一般来说，系统默认会启动 rpcbind ，不过鸟哥之前关闭过，所以要启动。
# 另外，如果服务器端有启动 nfslock 的话，客户端也要启动才能生效！

# 2\. 查询服务器提供哪些资源给我们使用呢？
[root@clientlinux ~]# showmount -e 192.168.100.254
Export list for 192.168.100.254:
/tmp         *
/home/linux  *.centos.vbird
/home/test   192.168.100.10
/home/public (everyone)   &lt;==这是等一下我们要挂载的目录
```

接下来我想要将远程主机的 /home/public 挂载到本地端主机的 /home/nfs/public ， 所以我就得要在本地端主机先建立起这个挂载点目录才行啊！然后就可以用 mount 这个指令直接挂载 NFS 的文件系统啰！

```
# 3\. 建立挂载点，并且实际挂载看看啰！
[root@clientlinux ~]# mkdir -p /home/nfs/public
[root@clientlinux ~]# mount -t nfs 192.168.100.254:/home/public \
&gt; /home/nfs/public
# 注意一下挂载的语法！『 -t nfs 』指定文件系统类型，
# IP:/dir 则是指定某一部主机的某个提供的目录！另外，如果出现如下错误：
mount: 192.168.100.254:/home/public failed, reason given by server: No such file
or directory
# 这代表你在 Server 上面并没有建立 /home/public 啦！自己在服务器端建立他吧！

# 4\. 总是得要看看挂载之后的情况如何，可以使用 df 或 mount 啦！
[root@clientlinux ~]# df
文件系统               1K-区段      已用      可用 已用% 挂载点
....(中间省略)....
192.168.100.254:/home/public
                       7104640    143104   6607104   3% /home/nfs/public
```

先注意一下挂载 NFS 档案的格式范例喔！呵呵！这样就可以将数据挂载进来啦！请注意喔！以后，只要你进入你的目录 /home/nfs/public 就等于到了 192.168.100.254 那部远程主机的 /home/public 那个目录中啰！很不错吧！至于你在该目录下有什么权限？ 那就请你回去前一小节查一查权限的思考吧！ ^_^ ！那么如何将挂载的 NFS 目录卸除呢？就使用 umount 啊！

```
[root@clientlinux ~]# umount /home/nfs/public
```

13.3.2 客户端可处理的挂载参数与开机挂载

瞧！客户端的挂载工作很简单吧！不过不晓得你有没有想过，如果你刚刚挂载到本机 /home/nfs/public 的文件系统当中，含有一支 script ，且这支 script 的内容为『 rm -rf / 』且该档案权限为 555 ， 夭寿～如果你因为好奇给他执行下去，可有的你受的了～因为整个系统都会被杀光光！真可怜！

所以说，除了 NFS 服务器需要保护之外，我们取用人家的 NFS 文件系统也需要自我保护才行啊！ 那要如何自我保护啊？可以透过 mount 的指令参数喔！包括底下这些主要的参数可以尝试加入：

参数	参数代表意义	系统默认值
suid nosuid	晓得啥是 SUID 吧？如果挂载的 partition 上面有任何 SUID 的 binary 程序时， 你只要使用 nosuid 就能够取消 SUID 的功能了！嗄？不知道什么是 SUID ？那就不要学人家架站嘛！@_@！ 赶紧回去基础学习篇第三版复习一下第十七章、程序与资源管理啦！	suid
rw ro	你可以指定该文件系统是只读 (ro) 或可擦写喔！服务器可以提供给你可擦写， 但是客户端可以仅允许只读的参数设定值！	rw
dev nodev	是否可以保留装置档案的特殊功能？一般来说只有 /dev 这个目录才会有特殊的装置，因此你可以选择 nodev 喔！	dev
exec noexec	是否具有执行 binary file 的权限？ 如果你想要挂载的仅是数据区 (例如 /home)，那么可以选择 noexec 啊！	exec
user nouser	是否允许使用者进行档案的挂载与卸除功能？ 如果要保护文件系统，最好不要提供使用者进行挂载与卸除吧！	nouser
auto noauto	这个 auto 指的是『mount -a』时，会不会被挂载的项目。 如果你不需要这个 partition 随时被挂载，可以设定为 noauto。	auto

一般来说，如果你的 NFS 服务器所提供的只是类似 /home 底下的个人资料， 应该不需要可执行、SUID 与装置档案，因此当你在挂载的时候，可以这样下达指令喔：

```
[root@clientlinux ~]# umount /home/nfs/public
[root@clientlinux ~]# mount -t nfs -o nosuid,noexec,nodev,rw \
&gt; 192.168.100.254:/home/public /home/nfs/public

[root@clientlinux ~]# mount &#124; grep addr
192.168.100.254:/home/public on /home/nfs/public type nfs (rw,noexec,nosuid,
nodev,vers=4,addr=192.168.100.254,clientaddr=192.168.100.10)
```

这样一来你所挂载的这个文件系统就只能作为资料存取之用，相对来说，对于客户端是比较安全一些的。 所以说，这个 nosuid, noexec, nodev 等等的参数可得记得啊！

- 关于 NFS 特殊的挂载参数

除了上述的 mount 参数之外，其实针对 NFS 服务器，咱们的 Linux 还提供不少有用的额外参数喔！这些特殊参数还非常有用呢！ 为什么呢？举例来说，由于文件系统对 Linux 是非常重要的东西，因为我们进行任何动作时，只要有用到文件系统， 那么整个目录树系统就会主动的去查询全部的挂载点。如果你的 NFS 服务器与客户端之间的联机因为网络问题， 或者是服务器端先关机了，却没有通知客户端，那么客户端只要动到文件系统的指令 (例如 df, ls, cp 等等) ，整个系统就会慢到爆！因为你必须要等到文件系统搜寻等待逾时后，系统才会饶了你！ (鸟哥等过 df 指令 30 分钟过...)

为了避免这些困扰，我们还有一些额外的 NFS 挂载参数可用！例如：

参数	参数功能	预设参数
fg bg	当执行挂载时，该挂载的行为会在前景 (fg) 还是在背景 (bg) 执行？ 若在前景执行时，则 mount 会持续尝试挂载，直到成功或 time out 为止，若为背景执行， 则 mount 会在背景持续多次进行 mount ，而不会影响到前景的程序操作。 如果你的网络联机有点不稳定，或是服务器常常需要开关机，那建议使用 bg 比较妥当。	fg
soft hard	如果是 hard 的情况，则当两者之间有任何一部主机脱机，则 RPC 会持续的呼叫，直到对方恢复联机为止。如果是 soft 的话，那 RPC 会在 time out 后『重复』呼叫，而非『持续』呼叫， 因此系统的延迟会比较不这么明显。同上，如果你的服务器可能开开关关，建议用 soft 喔！	hard
intr	当你使用上头提到的 hard 方式挂载时，若加上 intr 这个参数，则当 RPC 持续呼叫中，该次的呼叫是可以被中断的 (interrupted)。	没有
rsize wsize	读出(rsize)与写入(wsize)的区块大小 (block size)。 这个设定值可以影响客户端与服务器端传输数据的缓冲记忆容量。一般来说， 如果在局域网络内 (LAN) ，并且客户端与服务器端都具有足够的内存，那这个值可以设定大一点， 比如说 32768 (bytes) 等，提升缓冲记忆区块将可提升 NFS 文件系统的传输能力！ 但要注意设定的值也不要太大，最好是达到网络能够传输的最大值为限。	rsize=1024 wsize=1024

更多的参数可以参考 man nfs 的输出数据喔！ 通常如果你的 NFS 是用在高速运作的环境当中的话，那么可以建议加上这些参数的说：

```
[root@clientlinux ~]# umount /home/nfs/public
[root@clientlinux ~]# mount -t nfs -o nosuid,noexec,nodev,rw \
&gt; -o bg,soft,rsize=32768,wsize=32768 \
&gt; 192.168.100.254:/home/public /home/nfs/public
```

则当你的 192.168.100.254 这部服务器因为某些因素而脱机时，你的 NFS 可以继续在背景当中重复的呼叫！ 直到 NFS 服务器再度上线为止。这对于系统的持续操作还是有帮助的啦！ 当然啦，那个 rsize 与 wsize 的大小则需要依据你的实际网络环境而定喔！

Tips: 在鸟哥的实际案例中，某些大型的模式运算并不允许 soft 这个参数喔！举例来说，鸟哥惯用的 CMAQ 空气质量模式， 这个模式的丛集架构分享文件系统中，就不允许使用 soft 参数！这点需要特别留意喔！



- 将 NFS 开机即挂载

我们知道开机就挂载的挂载点与相关参数是写入 /etc/fstab 中的，那 NFS 能不能写入 /etc/fstab 当中呢？非常可惜的是， 不可以呢！为啥呢？分析一下开机的流程，我们可以发现网络的启动是在本机挂载之后，因此当你利用 /etc/fstab 尝试挂载 NFS 时，系统由于尚未启动网络，所以肯定是无法挂载成功的啦！那怎办？简单！就写入 /etc/rc.d/rc.local 即可！

```
[root@clientlinux ~]# vim /etc/rc.d/rc.local
mount -t nfs -o nosuid,noexec,nodev,rw,bg,soft,rsize=32768,wsize=32768 \
192.168.100.254:/home/public /home/nfs/public
```

13.3.3 无法挂载的原因分析

如果客户端就是无法挂载服务器端所分享的目录时，到底是发生什么问题？你可以这样分析看看：

- 客户端的主机名或 IP 网段不被允许使用：

以上面的例子来说明，我的 /home/test 只能提供 192.168.100.0/24 这个网域，所以如果我在 192.168.100.254 这部服务器中，以 localhost (127.0.0.1) 来挂载时，就会无法挂载上，这个权限概念没问题吧！不然你可以在服务器上试试看：

```
[root@www ~]# mount -t nfs localhost:/home/test /mnt
mount.nfs: access denied by server while mounting localhost:/home/test
```

看到 access denied 了吧？没错啦～权限不符啦！如果确定你的 IP 没有错误，那么请通知服务器端，请管理员将你的 IP 加入 /etc/exports 这个档案中。

- 服务器或客户端某些服务未启动：

这个最容易被忘记了！就是忘记了启动 rpcbind 这个服务啦！如果你在客户端发现 mount 的讯息是这样：

```
[root@clientlinux ~]# mount -t nfs 192.168.100.254:/home/test /mnt
mount: mount to NFS server '192.168.100.254' failed: System Error: Connection refused.
# 如果你使用 ping 却发现网络与服务器都是好的，那么这个问题就是 rpcbind 没有开啦！

[root@clientlinux ~]# mount -t nfs 192.168.100.254:/home/test /home/nfs
mount: mount to NFS server '192.168.100.254' failed: RPC Error: Program not registered.
# 注意看最后面的数据，确实有连上 RPC ，但是服务器的 RPC 告知我们，该程序无注册
```

要嘛就是 rpcbind 忘记开 (第一个错误)，要嘛就是服务器端的 nfs 忘记开。最麻烦的是， 重新启动了 rpcbind 但是却忘记重新启动其他服务 (上述第二个错误)！解决的方法就是去重新启动 rpcbind 管理的其他所有服务就是了！

- 被防火墙档掉了：

由于 NFS 几乎不对外开放，而内部网域又通常是全部的资源都放行，因此过去玩 NFS 的朋友 (包括鸟哥本人啦！) 都没有注意过 NFS 的防火墙问题。最近这几年鸟哥在管理计算机教室时，有掌管一部计算机教室主控防火墙， 为了担心太厉害的学生给鸟哥乱搞，因此该 Linux 防火墙预设是仅放行部分资源而已。但由于计算机教室的区网内需要用到 Linux 的 NFS 资源，结果呢？竟然没办法放行啊！原来就是 iptables 没有放行 NFS 所使用到的埠口～

所以，当你一直无法顺利的连接 NFS 服务器，请先到服务器端，将客户端的 IP 完全放行，若确定这样就连的上， 那代表就是防火墙有问题啦！怎么解决呢？上一小节介绍过了，参考将 NFS 服务器埠口固定的方式吧！

13.3.4 自动挂载 autofs 的使用

在一般 NFS 文件系统的使用情况中，如果客户端要使用服务器端所提供的 NFS 文件系统时，要嘛就是得在 /etc/rc.d/rc.local 当中设定开机时挂载，要嘛就得要登入系统后手动利用 mount 来挂载。 此外，客户端得要预先手动的建立好挂载点目录，然后挂载上来。但是这样的使用情况恐怕有点小问题。

- NFS 文件系统与网络联机的困扰：

我们知道 NFS 服务器与客户端的联机或许不会永远存在，而 RPC 这个服务又挺讨厌的，如果挂载了 NFS 服务器后，任何一方脱机都可能造成另外一方老是在等待逾时～而且，挂载的 NFS 文件系统可能又不是常常被使用，但若不挂载的话，有时候紧急要使用时又得通知系统管理员， 这又很不方便...啊！好讨厌的感觉啊～@_@

所以，让我们换个思考的角度来讨论一下使用 NFS 的情境：

- 可不可以让客户端在有使用到 NFS 文件系统的需求时才让系统自动挂载？
- 当 NFS 文件系统使用完毕后，可不可以让 NFS 自动卸除，以避免可能的 RPC 错误？

如果能达到上述的功能，那就太完美啦！有没有这东西呢？有的，在现在的 Linux 环境下这是可以达成的理想！用的就是 autofs 这个服务啦！

- autofs 的设定概念：

autofs 这个服务在客户端计算机上面，会持续的侦测某个指定的目录， 并预先设定当使用到该目录下的某个次目录时，将会取得来自服务器端的 NFS 文件系统资源，并进行自动挂载的动作。 讲这样或许你有点模糊，让我们拿底下这个图示来看看：

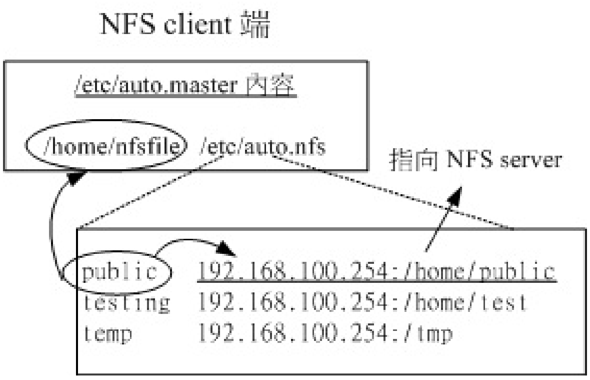


图 13.3-1、autofs 自动挂载的配置文件内容示意图

如上图所示，我们的 autofs 主要配置文件为 /etc/auto.master，这个档案的内容很简单， 如上所示，我只要定义出最上层目录 (/home/nfsfile) 即可，这个目录就是 autofs 会一直持续侦测的目录啦。 至于后续的档案则是该目录底下各次目录的对应。在 /etc/auto.nfs (这个档案的档名可自定义) 里面则可以定义出每个次目录所欲挂载的远程服务器的 NFS 目录资源！

举例来说：『当我们在客户端要使用 /home/nfsfile/public 的数据时，此时 autofs 才会去 192.168.100.254 服务器上挂载 /home/public ！』且『当隔了 5 分钟没有使用该目录下的数据后，则客户端系统将会主动的卸除 /home/nfsfile/public 』。

很不错用的一个工具吧！因为有用到服务器的数据时才自动挂载，没有使用了就会自动卸除！ 而不是传统的情况一直是挂载的！既然这么好用，那就让我们实际来操演一下：

- 建立主配置文件 /etc/auto.master ，并指定侦测的特定目录

这个主要配置文件的内容很简单，只要有要被持续侦测的目录及『数据对应文件』即可。 那个数据对应文件的文件名是可以自行设定的，在鸟哥这个例子当中我使用 /etc/auto.nfs 来命名。

```
[root@clientlinux ~]# vim /etc/auto.master
/home/nfsfile  /etc/auto.nfs
```

上述数据中比较需要注意的是，那个 /home/nfsfile 目录不需要存在，因为 autofs 会主动的建立该目录！ 如果你建立了，可能反而会出问题～因此，先确定一下没有该目录吧！

- 建立数据对应文件内 (/etc/auto.nfs) 的挂载信息与服务器对应资源

刚刚我们所指定的 /etc/auto.nfs 是自行设定的，所以这个档案是不存在的。那么这个档案的格式是如何呢？你可以这样看：

```
[本地端次目录]  [-挂载参数]  [服务器所提供的目录]
选项与参数：
[本地端次目录] ：指的就是在 /etc/auto.master 内指定的目录之次目录
[-挂载参数]     ：就是前一小节提到的 rw,bg,soft 等等的参数啦！可有可无；
[服务器所提供的目录] ：例如 192.168.100.254:/home/public 等

[root@clientlinux ~]# vim /etc/auto.nfs
public   -rw,bg,soft,rsize=32768,wsize=32768  192.168.100.254:/home/public
testing  -rw,bg,soft,rsize=32768,wsize=32768  192.168.100.254:/home/test
temp     -rw,bg,soft,rsize=32768,wsize=32768  192.168.100.254:/tmp
# 参数部分，只要最前面加个 - 符号即可！
```

这样就可以建立对应了！要注意的是，那些 /home/nfsfile/public 是不需要事先建立的！ 咱们的 autofs 会事情况来处理喔！好了，接下来让我们看看如何实际运作吧！

- 实际运作与观察

配置文件设定妥当后，当然就是要启动 autofs 啦！

```
[root@clientlinux ~]# /etc/init.d/autofs stop
[root@clientlinux ~]# /etc/init.d/autofs start
# 很奇怪！非常怪！CentOS 6.x 的 autofs 使用 restart 会失效！所以鸟哥才进行两次
```

假设你目前并没有挂载任何来自 192.168.100.254 这部 NFS 服务器的资源目录。好了， 那让我们实际来观察看看几个重要的数据吧！先看看 /home/nfsfile 会不会主动的被建立？ 然后，如果我要进入 /home/nfsfile/public 时，文件系统会如何变化呢？

```
[root@clientlinux ~]# ll -d /home/nfsfile
drwxr-xr-x. 2 root root 0 2011-07-28 00:07 /home/nfsfile
# 仔细看，妳会发现 /home/nfsfile 容量是 0 喔！那是正常的！因为是 autofs 建立的

[root@clientlinux ~]# cd /home/nfsfile/public
[root@clientlinux public]# mount &#124; grep nfsfile
192.168.100.254:/home/public on /home/nfsfile/public type nfs (rw,soft,rsize=32768,
wsize=32768,sloppy,vers=4,addr=192.168.100.254,clientaddr=192.168.100.10)
# 上面的输出是同一行！瞧！突然出现这个玩意儿！因为是自动挂载的嘛！

[root@clientlinux public]# df  /home/nfsfile/public
文件系统               1K-区段      已用      可用  已用% 挂载点
192.168.100.254:/home/public
                       7104640    143104   6607040   3% /home/nfsfile/public
# 档案的挂载也出现没错！
```

呵呵！真是好啊！如此一来，如果真的有需要用到该目录时，系统才会去相对的服务器上面挂载！ 若是一阵子没有使用，那么该目录就会被卸除呢！这样就减少了很多不必要的使用时机啦！还不错用吧！ ^_^

13.4 案例演练

让我们来做个实际演练，在练习之前，请将服务器的 NFS 设定数据都清除，但是保留 rpcbind 不可关闭。至于客户端的环境下， 先关闭 autofs 以及取消之前在 /etc/rc.d/rc.local 里面写入的开机自动挂载项目。同时删除 /home/nfs 目录吻！ 接下来请看看我们要处理的环境为何：

模拟的环境状态中，服务器端的想法如下：

1. 假设服务器的 IP 为 192.168.100.254 这一部；
2. /tmp 分享为可擦写，并且不限制使用者身份的方式，分享给所有 192.168.100.0/24 这个网域中的所有计算机；
3. /home/nfs 分享的属性为只读，可提供除了网域内的工作站外，向 Internet 亦提供数据内容；
4. /home/upload 做为 192.168.100.0/24 这个网域的数据上传目录，其中，这个 /home/upload 的使用者及所属群组为 nfs-upload 这个名字，他的 UID 与 GID 均为 210；
5. /home/andy 这个目录仅分享给 192.168.100.10 这部主机，以提供该主机上面 andy 这个使用者来使用，也就是说， andy 在 192.168.100.10 及 192.168.100.254 均有账号，且账号均为 andy ，所以预计开放 /home/andy 给 andy 使用他的家目录啦！

服务器端设定的实地演练：

好了，那么请你先不要看底下的答案，先自己动笔或者直接在自己的机器上面动手作作看，等到得到你要的答案之后， 再看底下的说明吧！

1. 首先，就是要建立 /etc/exports 这个档案的内容啰，你可以这样写吧！

```
[root@www ~]# vim /etc/exports
/tmp         192.168.100.0/24(rw,no_root_squash)
/home/nfs    192.168.100.0/24(ro)  *(ro,all_squash)
/home/upload 192.168.100.0/24(rw,all_squash,anonuid=210,anongid=210)
/home/andy   192.168.100.10(rw)
```

2. 再来，就是要建立每个对应的目录的实际 Linux 权限了！我们一个一个来看：

```
# 1\. /tmp
[root@www ~]# ll -d /tmp
drwxrwxrwt. 12 root root 4096 2011-07-27 23:49 /tmp

# 2\. /home/nfs
[root@www ~]# mkdir -p /home/nfs
[root@www ~]# chmod 755 -R /home/nfs
# 修改较为严格的档案权限将目录与档案设定成只读！不能写入的状态，会更保险一点！

# 3\. /home/upload
[root@www ~]# groupadd -g 210 nfs-upload
[root@www ~]# useradd -g 210 -u 210 -M nfs-upload
# 先建立对应的账号与组名及 UID 喔！
[root@www ~]# mkdir -p /home/upload
[root@www ~]# chown -R nfs-upload:nfs-upload /home/upload
# 修改拥有者！如此，则用户与目录的权限都设定妥当啰！

# 4\. /home/andy
[root@www ~]# useradd andy
[root@www ~]# ll -d /home/andy
drwx------. 4 andy andy 4096 2011-07-28 00:15 /home/andy
```

这样子一来，权限的问题大概就可以解决啰！

3. 重新启动 nfs 服务：

```
[root@www ~]# /etc/init.d/nfs restart
```

4. 在 192.168.100.10 这部机器上面演练一下：

```
# 1\. 确认远程服务器的可用目录：
[root@clientlinux ~]# showmount -e 192.168.100.254
Export list for 192.168.100.254:
/home/andy   192.168.100.10
/home/upload 192.168.100.0/24
/home/nfs    (everyone)
/tmp         192.168.100.0/24

# 2\. 建立挂载点：
[root@clientlinux ~]# mkdir -p /mnt/{tmp,nfs,upload,andy}

# 3\. 实际挂载：
[root@clientlinux ~]# mount -t nfs 192.168.100.254:/tmp         /mnt/tmp
[root@clientlinux ~]# mount -t nfs 192.168.100.254:/home/nfs    /mnt/nfs
[root@clientlinux ~]# mount -t nfs 192.168.100.254:/home/upload /mnt/upload
[root@clientlinux ~]# mount -t nfs 192.168.100.254:/home/andy   /mnt/andy
```

整个步骤大致上就是这样吶！加油喔！

13.5 重点回顾

- Network FileSystem (NFS) 可以让主机之间透过网络分享彼此的档案与目录；
- NFS 主要是透过 RPC 来进行 file share 的目的，所以 Server 与 Client 的 RPC 一定要启动才行！
- NFS 的配置文件就是 /etc/exports 这个档案；
- NFS 的权限可以观察 /var/lib/nfs/etab，至于的重要登录档可以参考 /var/lib/nfs/xtab 这个档案，还包含相当多有用的信息在其中！
- NFS 服务器与客户端的使用者账号名称、UID 最好要一致，可以避免权限错乱：
- NFS 服务器预设对客户端的 root 进行权限压缩，通常压缩其成为 nfsnobody 或 nobody。
- NFS 服务器在更动 /etc/exports 这个档案之后，可以透过 exportfs 这个指令来重新挂载分享的目录！
- 可以使用 rpcinfo 来观察 RPC program 之间的关系！！！
- NFS 服务器在设定之初，就必须要考虑到 client 端登入的权限问题，很多时候无法写入或者无法进行分享，主要是 Linux 实体档案的权限设定问题所致！
- NFS 客户端可以透过使用 showmount, mount 与 umount 来使用 NFS 主机提供的分享的目录！
- NFS 亦可以使用挂载参数，如 bg, soft, rsize, wsize, nosuid, noexec, nodev 等参数， 来达到保护自己文件系统的目标！
- 自动挂载的 autofs 服务可以在客户端需要 NFS 服务器提供的资源时才挂载。

13.6 本章习题

- NFS 的主要配置文件为何？而在该档案内主要设定项目为何？主要的配置文件为 /etc/exports 而至于其设定的内容项目在每一行当中则为：

  1. 分享的目录
  2. 针对此分享目录开放的主机或 IP或网域
  3. 针对这部主机所开放的权限参数！
- 在 NFS 主要的配置文件当中仅有少许的参数说明，至于预设的参数说明则没有在该档案当中出现，请问，如果要查阅更详细的分享出来的档案的属性，要看那个档案？/var/lib/nfs/etab
- 在 client 端如果要挂载 NFS 所提供分享的档案，可以使用那个指令？那自然就是 mount 啦！还有卸除是 umount 喔！
- 在 NFS 主要配置文件当中，可以透过那个参数来控制不让 client 端以 root 的身份使用你所分享出来的目录与档案？可以在 /etc/exports 当中的参数项目，设定『 root_squash 』来控制压缩 root 的身份喔！
- 我在 client 端挂载了 NFS Server 的某个目录在我的 /home/data 底下，当我执行其中某个程序时，却发现我的系统被破坏了？你认为可能的原因为何？ 该如何克服这样的问题，尤其是当我的 Client 端主机其实是多人共享的环境，怕其他的使用者也同样发生类似的问题呢？！

  - 可能由于你挂载进来的 NFS Server 的 partition 当中具有 SUID 的文件属性，而你不小心使用了该执行档，因此就可能会发生系统被破坏的问题了！
  - 可以将挂载进来的 NFS 目录的 SUID 功能取消！例如：
  - 可能由于你挂载进来的 NFS Server 的 partition 当中具有 SUID 的文件属性，而你不小心使用了该执行档，因此就可能会发生系统被破坏的问题了！
  - 可以将挂载进来的 NFS 目录的 SUID 功能取消！例如： mount -t nfs -o nosuid,ro server:/directory /your/directory

13.7 参考数据与延伸阅读

- 注1：Sun (升阳) 公司已经被甲骨文 (Oracle) 公司合并了，因此公司网址改于：http://www.oracle.com/us/sun/index.html
- http://www.faqs.org/rfcs/rfc1094.html 鸟哥这里的备份：http://linux.vbird.org/linux_server/0330nfs/0330nfs_rpc.html
- http://www.tldp.org/HOWTO/NFS-HOWTO/index.html
- man exports
- man autofs

2002/11/17：第一次完成 2003/03/09：修改部分内容，并且新增 LPI 相关性与重点整理部分！ 2003/09/10：又重新修改版面，以及新增主机的规划等部分。 2006/09/19：将旧的文章移动到 此处 2006/09/22：加入了 autofs 等等的数据喔！ 2007/02/27：原本 rsize 定义为 8192，但最近看一些文献，应该改为 32768 比较妥当。 2009/07/04：在最后的案例演练部分， IP 不可使用星号 (*) 的通配符！感谢讨论区网友acer07的回报！ 2011/03/03：将旧的基于 CentOS 4.x 的文章移动到 此处 2011/03/12：修订完成了！比较大的问题在于 nfsnobody 可能有时并不会压缩喔！ 2011/07/27：将基于 CentOS 5.x 的文章移动到此处

第十四章、账号控管： NIS 服务器

最近更新日期：2011/07/28

有没有想过，如果我有十部 Linux 主机，这十部主机仅负责不同的功能，事实上，所有的主机账号与对应的密码都相同！ 那么我是将账号与密码分别设定置在十部计算机上面，还是可以透过一部主机做为账号管理的功能，然后其他的主机只要当用户用登入时， 就必须要到管理账号的主机上面确认其账号与密码呢？哪一个比较方便而且灵活？当然是找一个账号管理的主机比较方便的多啦！ 如果有用户要修改密码，不必要去到十部主机修改密码啦！只要到主要管理主机去修改，其他的主机根本就不需要更动！ 哈哈！轻松又愉快呢！这个功能的达成有很多的方式，在这里，我们介绍一个很简单的方式，那就是 Network Information Service 这个 NIS 服务器的架设啦！

14.1 NIS 的由来与功能

在一个大型的网域当中，如果有多部 Linux 主机，万一要每部主机都需要设定相同的账号与密码时，你该怎么办？复制 /etc/passwd ？应该没有这么呆吧？如果能够有一部账号主控服务器来管理网域中所有主机的账号， 当其他的主机有用户登入的需求时，才到这部主控服务器上面要求相关的账号、密码等用户信息， 如此一来，如果想要增加、修改、删除用户数据，只要到这部主控服务器上面处理即可， 这样就能够降低重复设定使用者账号的步骤了。

这样的功能有很多的服务器软件可以达成，这里我们要介绍的则是 Network Information Services (NIS server) 这个服务器软件喔！底下就先来谈一谈这个 NIS 的相关功能吧！

Tips: NIS 主要提供的是用户的账号、密码、家目录文件名、UID等信息，但 NIS 并没有提供文件系统。同时， NIS 同样使用前一章谈到的 RPC 服务器，因此在本章开始前，你还是得要认识一下第十三章谈到的 NFS 与 RPC， 同时你还得要知道基础学习篇第三版里面的第十四章账号管理， 同时也得了解一下基础学习篇第二十二章 make/Makefile 的信息才好。



14.1.1 NIS 的主要功能：管理帐户信息

通常我们都会建议，一部 Linux 主机的功能越单纯越好，也就是说，一部 Linux 就专门进行一项服务。这样有许多的好处，这包含功能单纯所以系统资源得以完整运用， 并且在发生入侵或者是系统产生状况的时候，也比较容易追查问题所在。因此，一个公司内部常常会有好几部 Linux 主机，有的专门负责 WWW 、有的专门负责 Mail 、有的专门负责 SAMBA 等等的服务。

不过，这样虽然有分散风险、容易追踪问题的好处，但是，由于是同一个公司内的多部主机，所以事实上所有的 Linux 主机的账号与密码都是一样的！哇！那如果公司里面有 100 的人的话， 我们就需要针对这么多部的主机去设定账号密码了！而且，如果未来还有新进员工的话， 那么光是设定密码就会使系统管理员抓狂了！

这个时候，让我们换一个角度来思考：如果我设计了一部专门管理账号与密码的服务器，而其他的 Linux 主机当有客户端要登入的时候，就必须要到这部管理密码的服务器来查寻用户的账号与密码， 如此一来，我要管理所有的 Linux 主机的账号与密码，只要到那部主服务器上面去进行设定即可！ 包括新进人员的设定，反正其他的 Linux 主机都是向它查寻的嘛！没错！真是好～这个就是 Network Information Service, NIS 服务器的主要功能啦！

事实上，Network Information Service 最早应该是称为 Sun Yellow Pages (简称 yp)，也就是 Sun 这家公司出的一个名为 Yellow Pages 的服务器软件，请注意， NIS 与 YP 是一模一样的咚咚喔！这个 Yellow Pages 名字取的真是好！怎么说呢？知道黄页 (Yellow Pages) 是什么吗？就是我们家里的电话簿啦！ 今天如果你要查寻一家厂商的电话号码，通常就是直接去查黄页上面的纪录来取得电话号码啊！而这个 NIS 也一样，当使用者要登入时， Linux 系统就会到 NIS 服务器上面去找寻这个使用的账号与密码信息来加以比对， 以提供使用者登入之用的检验啊！很棒吧！ ^_^

那么 NIS 服务器提供了哪些信息呢？还记得账号与密码放置在哪里吧？NIS 就是提供那些数据啦！ 主要有底下这些基本的数据提供给有登入需求的主机喔：

服务器端文件名	档案内容
/etc/passwd	提供用户账号、UID、GID、家目录所在、Shell 等等
/etc/group	提供群组数据以及 GID 的对应，还有该群组的加入人员
/etc/hosts	主机名与 IP 的对应，常用于 private IP 的主机名对应
/etc/services	每一种服务 (daemons) 所对应的埠口 (port number)
/etc/protocols	基础的 TCP/IP 封包协定，如 TCP, UDP, ICMP 等
/etc/rpc	每种 RPC 服务器所对应的程序号码
/var/yp/ypservers	NIS 服务器所提供的数据库

至少可以提供上述这些功能，当然啦，你也可以自行定义哪些数据库需要，哪些数据库不需要！

14.1.2 NIS 的运作流程：透过 RPC 服务

由于 NIS 服务器主要是提供用户登入的信息给客户端主机来查询之用，所以， NIS 服务器所提供的数据当然就需要用到传输与读写比较快速的 "数据库" 文件系统， 而不是传统的纯文本数据。为了要达到这个目的，所以 NIS 服务器就必须要将前一小节提到的那些档案制作成为数据库档案， 然后使用网络协议让客户端主机来查询啰。至于所使用的通讯协议与前一章的 NFS 相同，都使用远程过程调用 (RPC) 这个玩意儿喔！

此外，如果在一个很大型的网域里面，万一所有的 Linux 主机都向同一部 NIS 服务器要求用户数据时， 这部 NIS 服务器的负载 (loading) 可能会过大。甚至如果考虑到数据使用的风险， 要是这单一的一部 NIS 服务器挂点时，那其他的 Linux 主机还要不要让 users 登入啊？所以啰，在较为大型的企业环境当中， NIS 服务器可以使用 master/slave (主控/辅助服务器) 架构的。

Master NIS 服务器提供系统管理者制作的数据库， slave 则取得来自 master 的数据，并藉以提供其他客户端的查询。 客户端可以向整个网域要求用户资料的响应，master 与 slave 皆可回答， 由于 slave 的数据来自于 master ，所以用户账号数据本身是同步的！ 如此一方面可以分散 NIS 服务器的负载，而且也可以避免因 NIS 服务器挂点而导致的无法登入的风险。

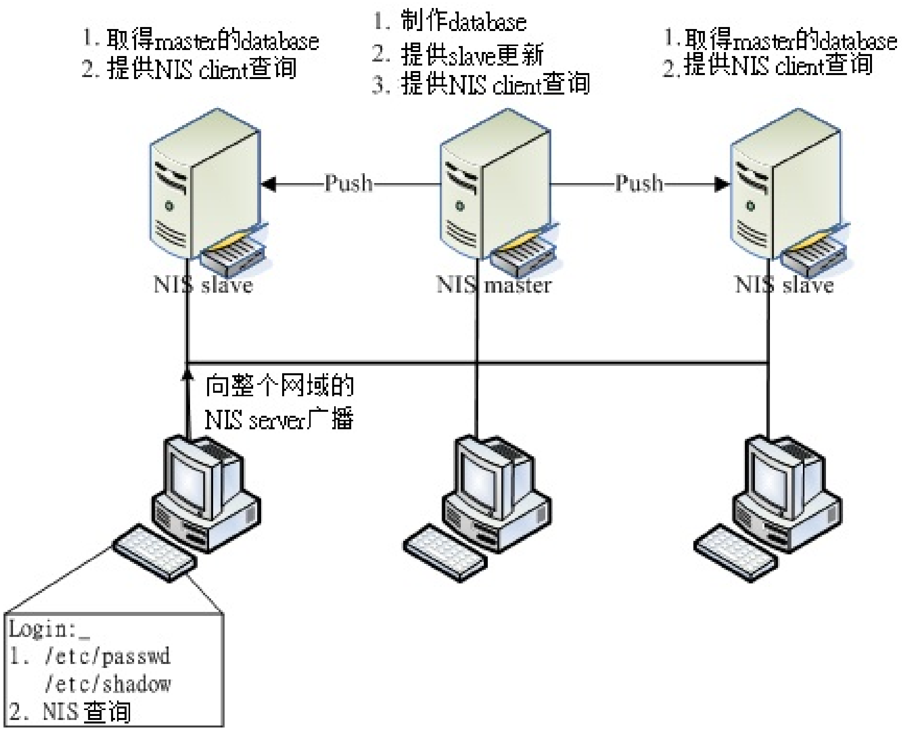


图 14.1-1、NIS 服务器与客户端的运作与查询方式示意图

整个 NIS 的运作就如同上图，首先必须要有 NIS server 的存在，之后才会有 NIS Client 的存在。 那么当使用者有登入的需求时，整个 NIS 的运作程序是：

- 关于 NIS Server (master/slave) 的运作程序：
  1. NIS Master 先将本身的账号密码相关档案制作成为数据库档案；
  2. NIS Master 可以主动的告知 NIS slave server 来更新；
  3. NIS slave 亦可主动的前往 NIS master server 取得更新后的数据库档案；
  4. 若有账号密码的异动时，需要重新制作 database 与重新同步化 master/slave。
- 关于当 NIS Client 有任何登入查询的需求时：
  1. NIS client 若有登入需求时，会先查询其本机的 /etc/passwd, /etc/shadow 等档案；
  2. 若在 NIS Client 本机找不到相关的账号数据，才开始向整个 NIS 网域的主机广播查询；
  3. 每部 NIS server (不论 master/slave) 都可以响应，基本上是『先响应者优先』。

从上面的流程当中，你会发现 NIS client 还是会先针对本机的账号数据进行查询，若本机查不到时才到 NIS server 上头寻找。因此，如果你的 NIS client 本身就有很多一般使用者的账号时，那跟 NIS server 所提供的账号就可能产生一定程度的差异啰！所以，一般来说，在这样

的环境下，NIS client 或 NIS slave server 会主动拿掉自己本机的一般使用者账号，仅会保留系统所需要的 root 及系统账号而已。 如此一来，一般使用者才都会经由 NIS master server 所控管啊！ ^_^

根据上面图 14.1-1 的说明，我们的 NIS 环境大致上需要设定的基本组件就有：

- NIS Master server ：将档案建置成数据库，并提供 slave server 来更新；
- NIS Slave server ：以 Master server 的数据库作为本身的数据库来源；
- NIS client ：向 master/server 要求登入者的验证数据。

就如同上面提到的，在大型环境中才会使用到这么复杂的 NIS master/slave 架构。因此，本章仅会介绍 NIS Master 的建置， 以及 NIS client 的设定而已。其实，NIS 服务使用的环境大概越来越仅局限在学术数值模式仿真的丛集计算机架构中 (PC cluster)， 在那样的架构中，老实说，鸟哥认为仅要学会 NIS master 即可。如果还有其他账号方面的要求，例如跨平台的帐户信息提供， 那可能就得要参考 Samba 或更进阶的 LDAP 才好哟！这里我们不谈啦～现在，就让我们开始来玩一玩这个 NIS 的设定吧！

14.2 NIS Server 端的设定

NIS 服务器端主要在于提供数据库给客户端作为验证之用，虽然 NIS 服务器类型有 Master 与 Slave ， 不过鸟哥这里介绍的并不是大型企业环境，因此仅介绍 NIS master 的设定而已啦～ 那就来设定看看啰！

14.2.1 所需要的软件

由于 NIS 服务器需要使用 RPC 协议，且 NIS 服务器同时也可以当成客户端，因此它需要的软件就有底下这几个：

- yp-tools ：提供 NIS 相关的查寻指令功能
- ypbind ：提供 NIS Client 端的设定软件
- ypserv ：提供 NIS Server 端的设定软件
- rpcbind ：就是 RPC 一定需要的数据啊！

如果你是使用 Red Hat 的系统，例如我们的 CentOS 6.x 的话，那你可以利用『 rpm -qa | grep '^yp' 』 来检查是否有安装上述的软件。一般来说 yp-tools, ypbind 都会主动的安装，不过 ypserv 可能就不会安装了。 此时建议你直接使用『 yum install ypserv 』来安装吧！立刻就装好了。底下立刻来设定啰！

14.2.2 NIS 服务器相关的配置文件

在 NIS 服务器上最重要的就是 ypserv 这个软件了，但是，由于 NIS 设定时还会使用到其他网络参数设定数据， 因此在配置文件方面需要有底下这些数据喔：

- /etc/ypserv.conf：这是最主要的 ypserv 软件所提供的配置文件，可以规范 NIS 客户端是否可登入的权限。
- /etc/hosts：由于 NIS server/client 会用到网络主机名与 IP 的对应，因此这个主机名对应档就显的相当重要！每一部主机名与 IP 都需要记录才行！
- /etc/sysconfig/network：可以在这个档案内指定 NIS 的网域 (nisdomainname)。
- /var/yp/Makefile：前面不是说账号数据要转成数据库文件吗？ 这就是与建立数据库有关的动作配置文件；

至于 NIS 服务器提供的主要服务方面有底下两个：

- /usr/sbin/ypserv：就是 NIS 服务器的主要提供服务；
- /usr/sbin/rpc.yppasswdd：提供额外的 NIS 客户端之用户密码修改服务， 透过这个服务， NIS 客户端可以直接修改在 NIS 服务器上的密码。相关的使用程序则是 yppasswd

指令；

与账号密码的数据库有关的指令方面有底下几个：

- /usr/lib64/yp/ypinit：建立数据库的指令，非常常用 (在 32 位的系统下，文件名则是 /usr/lib/yp/ypinit 喔！)；
- /usr/bin/yppasswd：与 NIS 客户端有关，主要在让用户修改服务器上的密码。

14.2.3 一个实作案例

如果你有观察过图 14.1-1 的话，你会发现到我们的 NIS 需要设定 Master/Slave 及 client 等，不过我们这里仅介绍 NIS master server 与 NIS client 两个组件而已，如果你有需要额外的 slave 的话， 再请查阅 NIS 官网的介绍啰。底下鸟哥先拟一个简单的案例，做完案例我们再来谈谈实际可能会使用于丛集计算机的案例吧！

- NIS 的域名为 vbirdnis
- 整个内部的信任网域为 192.168.100.0/24
- NIS master server 的 IP 为 192.168.100.254 ，主机名为 www.centos.vbird
- NIS client 的 IP 为 192.168.100.10，主机名为 clientlinux.centos.vbird

底下我们就一个一个来设定吧！

14.2.4 NIS server 的设定与启动

NIS 服务器的设定真是很简单，首先，你必须要在 NIS 服务器上面搞定你的账号与密码相关数据， 这包括 /etc/passwd, /etc/shadow, /etc/hosts, /etc/group .... 等等，都得要先搞定才行！ 详细的账号相关资料请参考基础篇的第十四章账号管理。 等到搞定之后你就可以继续 NIS 服务器的设定了：

- 1. 先设定 NIS 的域名 (NIS domain name)

NIS 是会分领域名 (domain name) 来分辨不同的账号密码数据的，因此你必须要在服务器与客户端都指定相同的 NIS 领域名才行。设定这个 NIS 领域名的动作很简单，就直接编辑 /etc/sysconfig/network 即可！如下所示：

```
[root@www ~]# vim /etc/sysconfig/network
# 不要更改其他既有数据，只要加入底下这几行即可：
NISDOMAIN=vbirdnis       &lt;==设定 NIS 领域名
YPSERV_ARGS="-p 1011"    &lt;==设定 NIS 每次都启动在固定的埠口
```

当然，你也可以使用手动的方式暂时设定好你的 NIS 领域名，透过的方法就是 nisdomainname 这个指令。 (其实 nisdomainname 与 ypdomainname 及 domainname 都是一模一样的指令啦！你只要记住一个指令名称即可。请自行 man domainname 吧！)不过，这个指令现在大概只用来检查设定是否正确，因为启动 NIS 服务器时，服务器去捉取的数据就是从 network 这个档案里面捉取的！所以只要改这个配置文件即可啊！

另外，由于未来想使用 iptables 直接管理 NIS 的使用，因此我们想要控制 NIS 启动在固定的埠口上。此时， 就使用『YPSERV_ARGS="-p 1011"』这个设定值来固定埠口在 1011 吧！

- 2. 主要配置文件 /etc/ypserv.conf

这个配置文件就是 NIS 服务器最主要的配置文件啦！内容其实很简单，你可以保留默认值即可。不过，也可以作一些更动啦！

```
[root@www ~]# vim /etc/ypserv.conf
dns: no
# NIS 服务器大多使用于内部局域网络，只要有 /etc/hosts 即可，不用 DNS 啦

files: 30
# 预设会有 30 个数据库被读入内存当中，其实我们的账号档案并不多，30 个够用了。

xfr_check_port: yes
# 与 master/slave 有关，将同步更新的数据库比对所使用的端口，放置于 &lt;1024 内。

# 底下则是设定限制客户端或 slave server 查询的权限，利用冒号隔成四部分：
# **[主机名/IP] : [NIS域名] : [可用数据库名称] : [安全限制]**
# [主机名/IP]   ：可以使用 network/netmask 如 192.168.100.0/255.255.255.0
# [NIS域名]    ：例如本案例中的 vbirdnis
# [可用数据库名称]：就是由 NIS 制作出来的数据库名称；
# [安全限制]      ：包括没有限制 (none)、仅能使用 &lt;1024 (port) 及拒绝 (deny)
# 一般来说，你可以依照我们的网域来设定成为底下的模样：
127.0.0.0/255.255.255.0     : * : * : none
192.168.100.0/255.255.255.0 : * : * : none
*                           : * : * : deny
# 星号 (*) 代表任何数据都接受的意思。上面三行的意思是，开放 lo 内部接口、
# 开放内部 LAN 网域，且杜绝所有其他来源的 NIS 要求的意思。

# 还有一个简单作法，你可以先将上面三行批注，然后加入底下这一行即可：
*                         : * : * : none
```

由于鸟哥习惯在内部网域并不设定比较严格的限制，因此通常鸟哥都是选择使用『 : : * : none 』那个设定值！ 然后透过 iptables 来管控可使用的来源就是了。当然，你可以依据你的需求来设定啦！

- 3. 设定主机名与 IP 的对应 (/etc/hosts)

在 /etc/ypserv.conf 的设定当中我们谈到 NIS 大部分是给局域网络内的主机使用的，所以当然就不需要 DNS 的设定了。不过，由于 NIS 使用到很多的主机名，但是网络联机透过的是 IP 啊！所以你一定要设定好 /etc/hosts 里面的主机名与 IP 的对应，否则会无法成功联机 NIS ！这个很重要，绝大部分的朋友无法达成 NIS server/client 的联机都是这里出问题而已。 依据本案例的设定值，你应该这样做：

```
[root@www ~]# vim /etc/hosts
# 原本就有的 localhost 与 127.0.0.1 之类的设定都不要更动，只要新增数据：
&lt;u&gt;192.168.100.254   www.centos.vbird&lt;/u&gt;
192.168.100.10    clientlinux.centos.vbird

[root@www ~]# hostname
www.centos.vbird
# 再做个确认，确定输出的主机名与本机 IP 确实有写入 /etc/hosts 喔！
```

注意！如果你的主机名 (hostname) 与 NIS 的主机名不一样，那么在这个档案当中还是需要将你的主机名给他设定进来！ 否则在后面数据库的设定时，肯定会发生问题。当然啦，你也可以直接在 /etc/sysconfig/network 当中直接重新设定主机名，然后重新启动，或者是利用 hostname 这个指令重新设定你的主机名也可以。

- 4. 启动与观察所有相关的服务

接下来当然是先启动所有相关的服务啰，这包括 RPC, ypserv 以及 yppasswdd 啰！不过，如果你的 RPC 本来就已经启动的话，那就不要重新启动 rpcbind 了！此外，为了也让 yppasswdd 启动在固定的埠口，方便防火墙的管理， 因此，我们也建议你可以设定一下 /etc/sysconfig/yppasswdd 喔！

```
[root@www ~]# vim /etc/sysconfig/yppasswdd
YPPASSWDD_ARGS="--port  1012"    &lt;==找到这个设定值，修改一下内容成这样！

[root@www ~]# /etc/init.d/ypserv start
[root@www ~]# /etc/init.d/yppasswdd start
[root@www ~]# chkconfig ypserv on
[root@www ~]# chkconfig yppasswdd on
```

注意，主要的 NIS 服务是 ypserv ，不过，如果要提供 NIS 客户端的密码修改功能的话， 最好还是得要启动 yppasswdd 这个服务才好。在启动完毕后，我们可以利用 rpcinfo 来检查看看：

```
[root@www ~]# rpcinfo -p localhost
   program vers proto   port  service
    100000    4   tcp    111  portmapper
    100000    4   udp    111  portmapper
    100004    2   udp   1011  ypserv
    100004    1   udp   1011  ypserv
    100004    2   tcp   1011  ypserv
    100004    1   tcp   1011  ypserv
    100009    1   udp   1012  yppasswdd
# 其他不相干的 RPC 鸟哥将它拿掉了，与 NIS 有关的至少要有上面这几个！要仔细看，
# 看看埠口是否为我们规定的 1011, 1012 ，若不是的话，得要修改一下配置文件。

[root@www ~]# rpcinfo -u localhost ypserv
program 100004 version 1 ready and waiting
program 100004 version 2 ready and waiting
```

很多时候，很多朋友在设定完 NIS 后又回去设定 NFS 了，结果看了前一章的介绍，竟然又重新启动 rpcbind ， 这将导致 ypserv 的注册数据被注销掉。因此，使用上述的动作来检查看看服务有没有在等待中， 要看到如上的『就绪并等待服务』才会是正常的呦！

- 5. 处理账号并建立数据库

在完成了上面的所有步骤后，接下来我们得要开始将主机上面的账号档案转成数据库档案啦！ 不过，因为担心与 NIS 客户端的账号有冲突，加上之前我们已经建立过一些账号了。所以，这里我们建立三个新账号， 分别是 nisuser1, nisuser2, nisuser3 。不过账号主要是依据 UID 来判断的啊！因此，我们使用大于 1000 的 UID 来建立这三个账号喔！

```
[root@www ~]# useradd -u 1001 nisuser1
[root@www ~]# useradd -u 1002 nisuser2
[root@www ~]# useradd -u 1003 nisuser3
[root@www ~]# echo password &#124; passwd --stdin nisuser1
[root@www ~]# echo password &#124; passwd --stdin nisuser2
[root@www ~]# echo password &#124; passwd --stdin nisuser3
```

接下来，将建立的帐密数据转成数据库吧！转换的动作直接透过 /usr/lib64/yp/ypinit 这个指令来处理即可！整个步骤是这样做的：

```
[root@www ~]# /usr/lib64/yp/ypinit -m

At this point, we have to construct a list of the hosts which will run NIS
servers.  www.centos.vbird is in the list of NIS server hosts.  Please continue
to add the names for the other hosts, one per line.  When you are done with the
list, type a &lt;control D&gt;.
        next host to add:  www.centos.vbird  &lt;==系统根据主机名自动捉取
        next host to add:                    &lt;==这个地方按下 [crtl]-d
The current list of NIS servers looks like this:

www.centos.vbird

Is this correct?  [y/n: y]  y
We need a few minutes to build the databases...
Building /var/yp/vbirdnis/ypservers...
Running /var/yp/Makefile...
gmake[1]: Entering directory `/var/yp/vbirdnis'
Updating passwd.byname...
Updating passwd.byuid...
....(中间省略)....
gmake[1]: Leaving directory `/var/yp/vbirdnis'

www.centos.vbird has been set up as a NIS master server.

Now you can run ypinit -s www.centos.vbird on all slave server.
```

要注意出现的信息当中，在告知你可以直接输入 [ctrl]-d 以结束的那个地方， 你的主机名会主动的被捉出来，注意！这个主机名务必需要在 /etc/hosts 可以被找到 IP 的对应， 否则会出现问题。另外，万一在执行 ypinit -m 时，出现如下的错误，那肯定就是有些数据没有被建立了！

```
gmake[1]: *** No rule to make target `/etc/aliases', needed by `mail.aliases'.  Stop.
gmake[1]: Leaving directory `/var/yp/vbirdnis'
make: *** [target] Error 2
Error running Makefile.
Please try it by hand.

[root@www ~]# touch /etc/aliases
# 解决方法很简单呐！缺少什么档案，就 touch 他就是了！

[root@www ~]# /usr/lib64/yp/ypinit -m
# 然后再重新执行一次即可！
```

如果是如下的错误，那可能是因为：

- 你的 ypserv 服务没有顺利启动，请利用 rpcinfo 检查看看；
- 你的主机名与 IP 没有对应好，请检查 /etc/hosts

```
gmake[1]: Entering directory `/var/yp/vbirdnis'
Updating passwd.byname...
failed to send 'clear' to local ypserv: RPC: Program not registeredUpdating passwd.byuid.
failed to send 'clear' to local ypserv: RPC: Program not registeredUpdating group.byname.
....(底下省略)....
```

要注意啊，如果你的用户密码有变动过，那么你就得要重新制作数据库，重新启动 ypserv 及 yppasswdd 喔！ 注意注意啊！整个 NIS 服务器这样就给搞定了，有没有很简单啊！

14.2.5 防火墙设置

又来到了防火墙的规划了！要注意的是，我们的 NIS 与 NFS 都是使用 RPC Server 的，所以啰，除了上述谈到的固定埠口之外， 你还得要开放 port 111 才行。假设你已经看过前一章，而且是使用鸟哥的 iptables.rule 脚本来处理你的防火墙， 那么你可以修改该档案的内容，新增几条规则去：

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.allow
iptables -A INPUT -i $EXTIF -p tcp -s 192.168.100.0/24 --dport 1011 -j ACCEPT
iptables -A INPUT -i $EXTIF -p udp -s 192.168.100.0/24 -m multiport \
         --dport 1011,1012 -j ACCEPT

[root@www ~]# /usr/local/virus/iptables/iptables.rule
# 千万记得要重新建置防火墙规则啊！
```

14.3 NIS Client 端的设定

我们知道网络联机是双向的，所以 NIS server 提供数据库档案，NIS client 当然也需要提供一些联机的软件啰！这个联机的软件就是 ypbind 啦！此外，如同图 14.1-1 的介绍，在 NIS client 端有登入需求时，NIS client 基本上还是先搜寻自己的 /etc/passwd, /etc/group 等数据后才再去找 NIS server 的数据库啊！所以 NIS client 最好能够将本身的账号密码删除到仅剩下系统账号，亦即 UID, GID 均小于 500 以下的账号即可， 如此一来既可让系统执行无误，也能够让登入者的信息完全来自 NIS server ，比较单纯啦！

Tips: 事实上，你想要让 NIS 服务器写入的各项账号数据都在 NIS server 的 /var/yp/Makefile 那个档案设定的！ 你可以进入该档案搜寻一下 UID 就知道了！ ^_^



14.3.1 NIS client 所需软件与软件结构

NIS client 端所需要的软件仅有：

- ypbind
- yp-tools

yp-tools 是提供查询的软件，至于 ypbind 则是与 ypserv 互相沟通的客户端联机软件啦！ 另外，在 CentOS 当中我们还有很多配置文件是与认证有关的，包含 ypbind 的配置文件时， 在设定 NIS client 你可能需要动到底下的档案：

- /etc/sysconfig/network：就是 NIS 的领域名嘛！
- /etc/hosts：至少需要有各个 NIS 服务器的 IP 与主机名对应；
- /etc/yp.conf：这个则是 ypbind 的主要配置文件，里面主要设定 NIS 服务器所在
- /etc/sysconfig/authconfig：规范账号登入时的允许认证机制；
- /etc/pam.d/system-auth ：这个最容易忘记！因为账号通常由 PAM 模块所管理， 所以你必须要在 PAM 模块内加入 NIS 的支持才行！
- /etc/nsswitch.conf ：这个档案可以规范账号密码与相关信息的查询顺序，默认是先找 /etc/passwd 再找 NIS 数据库；

另外， NIS 还提供了几个有趣的程序给 NIS 客户端来进行账号相关参数的修改，例如密码、shell 等等， 主要有底下这几个指令：

- /usr/bin/yppasswd ：更改你在 NIS database (NIS Server 所制作的数据库) 的密码

- /usr/bin/ypchsh ：同上，但是是更改 shell
- /usr/bin/ypchfn ：同上，但是是更改一些用户的讯息！

OK！那么底下就让我们开始来设定 NIS 客户端吧！^_^

14.3.2 NIS client 的设定与启动

启动 NIS client 的设定就简单多了！最主要是加入 NIS domain 当中，然后再启动 ypbind 即可。 虽然你可以手动去修改所有的配置文件，然而近期以来的 Linux distributions 账号处理机制越来越复杂， 所以如果你想要手动修改所有配置文件，恐怕会疯掉的～因此，这里建议你使用系统提供的工具来设定， 至于一些重要配置文件，最后有机会再去参考一下即可。

那么 CentOS 6.x 提供了什么好用的管理工具呢？很简单，就利用 setup 这个指令即可！输入 setup 就会出现如下的图示， 然后依序这么处理就好了呦！

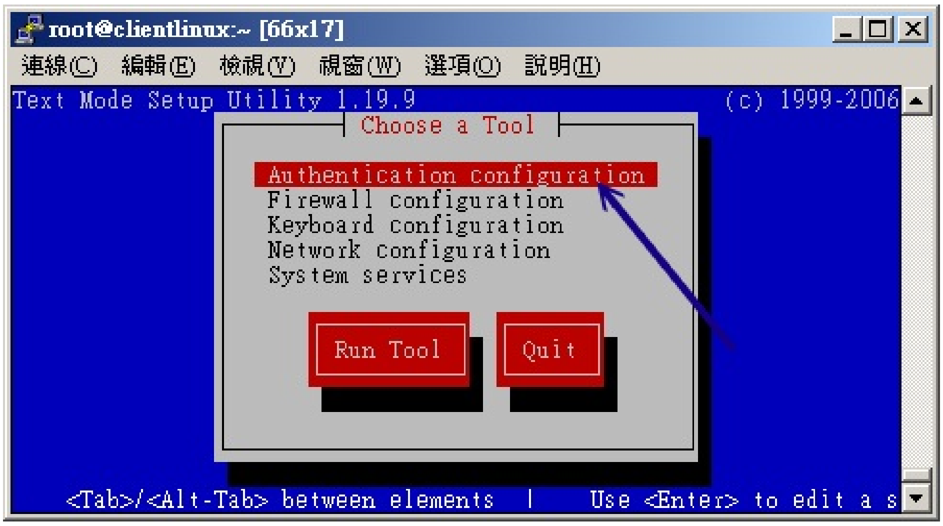


图 14.3-1、利用 setup 进入 authconfig 认证项目

记得在出现上图 14.3-1 后，选择认证设定，如果是出现英文的话，那么你就得要选择『Authentication configuration』的项目，之后就会进入下面的画面：

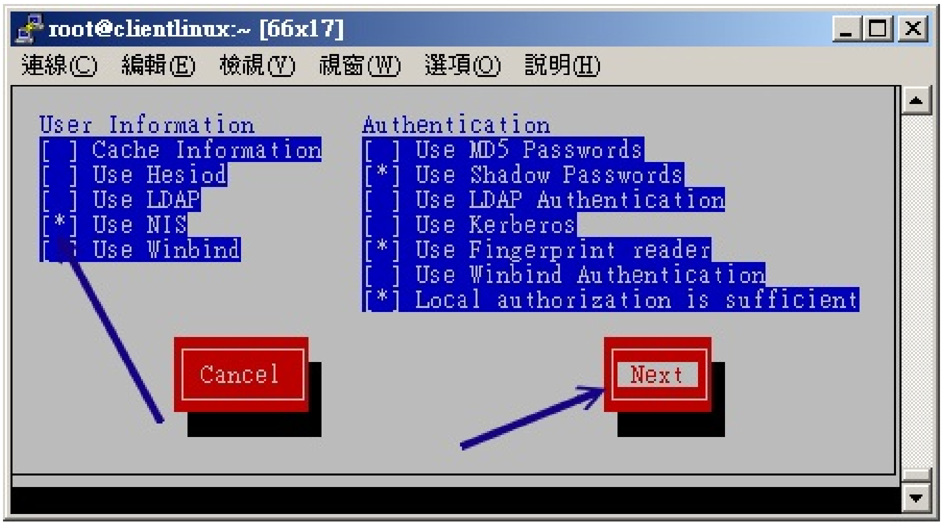


图 14.3-2、进入 authconfig 之后，选择 NIS 项目

因为我们要用 NIS 作为登入者身份验证的机制，因此就得要选择 NIS 项目，如果是英文的话，得要选择『Use NIS』项目即可。

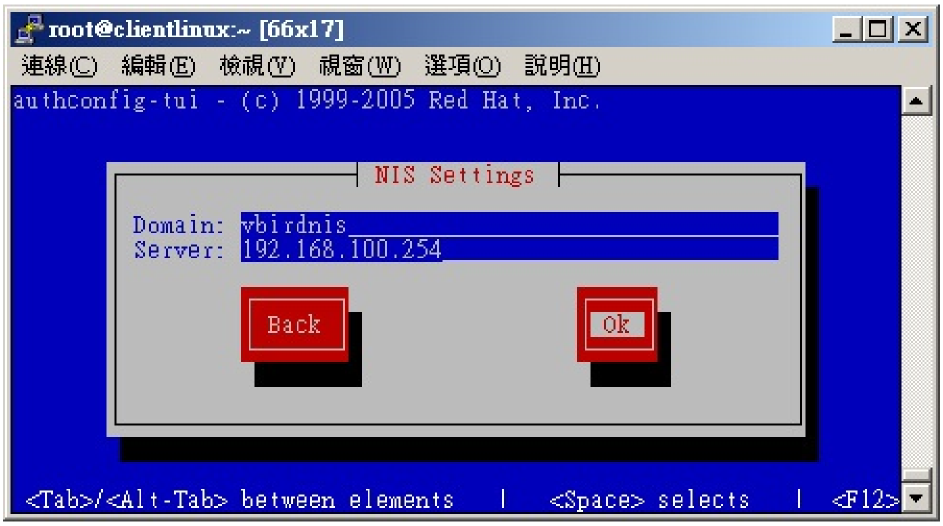


图 14.3-3、填写 NIS 领域以及 NIS 服务器的 IP 即可

最后再填写 NIS 网域 (Domain) 以及 NIS 服务器的 IP (Server)，按下确定即可。如果系统很快的就跳回图 14.3-1 的画面， 代表你的设定理论上是没有问题的。如果一直卡在如下的画面中：

```
正在激活 rpcbind：                                                [  确定  ]
正在关闭 NIS 服务：                                               [  确定  ]
正在启动 NIS 服务：                                               [  确定  ]
正在绑定 NIS 服务：.......  &lt;==这里一直卡住，没办法结束
```

上述的数据就是出问题啦！那代表你的 NIS client 没有办法连接上 NIS server，最常发生的就是服务器的防火墙忘记放行，或者是你客户端输入服务器 IP 时，打错数字了～ 那也是很常发生的错误啦！这时请自行去修改一番吧！那么这个 setup 到底做了什么修改呢？我们也来看看几个被改掉的重要配置文件吧：

```
[root@clientlinux ~]# cat /etc/sysconfig/network
HOSTNAME=clientlinux.centos.vbird
NETWORKING=yes
GATEWAY=192.168.100.254
NISDOMAIN=vbirdnis    &lt;==这个玩意儿会主动的被建立起来

[root@clientlinux ~]# cat /etc/yp.conf
....(前面省略)....
domain vbirdnis server 192.168.100.254  &lt;==主动建立这玩意儿～

[root@clientlinux ~]# vim /etc/nsswitch.conf
passwd:     files nis
shadow:     files nis
group:      files nis
hosts:      files nis dns
# 上面几个项目是比较重要的，包括身份参数、密码、群组名、主机名与 IP 对应数据等。
# 你会看到，每个项目后面都会接着 nis ，所以 nis 有被支持啰！
```

因为更动到的档案实在太多了，所以鸟哥还是建议使用 setup 来调整即可。但是，如果你真的想要手动处理的话， 那么你必须要手动的修改底下这些档案：

- /etc/sysconfig/network (加入 NISDOMAIN 项目)
- /etc/nsswitch.conf (修改许多主机验证功能的顺序)
- /etc/sysconfig/authconfig (CentOS 的认证机制)
- /etc/pam.d/system-auth (许多登入所需要的 PAM 认证过程)
- /etc/yp.conf (亦即是 ypbind 的配置文件)

14.3.3 NIS client 端的检验： yptest, ypwhich, ypcat

如何确定 NIS client 已经连上 NIS server 呢？基本上，只要刚刚使用 setup 去设定时，最后的步骤并没有被卡住， 那应该就是顺利成功啦！该步骤会自动启动 rpcbind 与 ypbind 两个服务哟！那如何确认数据传送是正确的？ 简单的要命啊！你可以利用 id 这个指令直接检查 NIS server 有的，但是 NIS client 没有的账号，如果有出现该账号的相关 UID/GID 信息时，那表示数据传输也是正确的。 除此之外，我们还可以透过 NIS 提供的相关检验功能来检查喔！底下分别来瞧一瞧：

- 利用 yptest 检验数据库之测试：

直接在 NIS client 输入 yptest 即可检查相关的测试数据，如下所示：

```
[root@clientlinux ~]# yptest
Test 1: domainname
Configured domainname is "vbirdnis"

Test 2: ypbind
Used NIS server: www.centos.vbird

Test 3: yp_match
WARNING: No such key in map (Map passwd.byname, key nobody)
....(中间省略)....

Test 6: yp_master
www.centos.vbird

....(中间省略)....

Test 8: yp_maplist
passwd.byname
protocols.byname
hosts.byaddr
hosts.byname
....(中间省略)....

Test 9: yp_all
nisuser1 nisuser1:$1$U9Gccb60$K5lDQ.mGBw9x4oNEkM0Lz/:1001:1001::/home/nisuser1:/bin/bash
....(中间省略)....
1 tests failed
```

从这个测试当中我们可以发现一些错误，就是在 Test 3 出现的那个警告信息啦。还好，那只是说没有该数据库而已～ 该错误是可以忽略的。重点在第 9 个步骤 yp_all 必须要有列出你 NIS server 上头的所有帐户信息，如果有出现账号相关数据的话，那么应该就算验证成功了！

Tips: 比较有问题的是第三步骤，他会出现在 passwd.byname 当中找不到 nobody 的字样。这是因为早期的 nobody 之 UID 都设定在 65534 ，但 CentOS 则将 nobody 设定为系统账号的 99 ，所以当然不会被记录，也就出现这一个警告。不过，这个错误是可忽略的啦！



- 利用 ypwhich 检验数据库数量

单纯使用 ypwhich 的时候显示的是『NIS Client 的 domain』名称，而当加入 -x 这个参数时，则是显示『NIS Client 与 Server 之间沟通的数据库有哪些？』你可以这样测试哩！

```
[root@clientlinux ~]# ypwhich -x
Use "hosts"     for map "hosts.byname"
Use "group"     for map "group.byname"
Use "passwd"    for map "passwd.byname"
....(以下省略)....
```

由上面我们可以很清楚的就看到相关的档案啦！这些数据库档案则是放置在我的 NIS Server 的 /var/yp/vbirdnis/* 里面啰！

- 利用 ypcat 读取数据库内容

除了 yptest 之外，你还可以直接利用 ypcat 读取数据库的内容喔！一般作法是这样：

```
[root@clientlinux ~]# ypcat [-h nisserver] [数据库名称]
选项与参数：
-h nisserver ：如果有设定的话，指向某一部特定的 NIS 服务器，
               如果没有指定的话，就以 ypbind 之设定为主；
数据库名称：亦即在 /var/yp/vbirdnis/ 内的档名啊！例如 passwd.byname

# 读出 passwd.byname 的数据库内容
[root@clientlinux ~]# ypcat passwd.byname
```

这三个指令在进行 NIS Client 端的检验时，是相当有用的喔！不要忽略了他的存在啊！尤其是刚架设好 NIS Client 时，一定要使用 yptest 去检查看看有没有设定错误喔！根据屏幕显示的讯息去一个一个校正错误才行啊！

14.3.4 使用者参数修改： yppasswd, ypchfn, ypchsh

好了，完成了上述的设定后，你的 NIS server/client 的账号已经同步了！真是高兴不是吗？不过，还有个挺大的问题，那就是...使用者如何在 NIS client 修改他自己的登入参数，例如密码、shell 等等？ 因为 NIS client 是藉由数据库来取得用户的账号密码，那如何在 NIS 客户端处理账号密码的订正？

问的好！这也是为何我们需要在 NIS server 启动 yppasswdd 这支服务的主要用意！ 因为 yppasswdd 可以接收 NIS client 端传来的密码修改，藉此而处理 NIS server 的 /etc/passwd, /etc/shadow ， 然后 yppasswdd 还能够重建密码数据库，让 NIS server 同步更新数据库！真是很不错啊！ ^_^

那该如何下达指令呢？很简单啊！透过 yppasswd, ypchsh, ypchfn 来处理即可。这三个指令的对应是：

- yppasswd ：与 passwd 指令相同功能；
- ypchfn ：与 chfn 相同功能；
- ypchsh ：与 chsh 相同功能。

因为功能相当，所以鸟哥这里仅说明一下 yppasswd 而已。假设你已经登入 NIS client 那部主机， 并且是以 nisuser1 这个使用者登入的，记住，这个用户相关数据仅在 NIS server 上。接下来，这个使用者可以下达 yppasswd ，如下所示：

```
[root@clientlinux ~]# grep nisuser /etc/passwd  &lt;==不会出现任何讯息，因为无此账号
[root@clientlinux ~]# su - nisuser1             &lt;==直接切换身份看看！
su: warning: cannot change directory to /home/nisuser1: No such file or directory
-bash-4.1$ id
uid=1001(nisuser1) gid=1001(nisuser1) groups=1001(nisuser1)
# 因为我们 client.centos.vbird 仅有帐户信息，并没有用户家目录，
# 所以就会出现如上的警告，因此才需要用 id 验证，并且需要加挂 NFS 嘛！
# 仔细看，现在的身份确实是 nisuser1 喔！确实有连上 NIS server 啦！

-bash-4.1$ yppasswd
Changing NIS account information for nisuser1 on www.centos.vbird.
Please enter old password:    &lt;==这里输入旧密码
Changing NIS password for nisuser1 on www.centos.vbird.
Please enter new password:    &lt;==这里输入新密码
Please retype new password:   &lt;==再输入一遍

The NIS password has been changed on www.centos.vbird.

-bash-4.1$ exit
```

嘿嘿！如何，这样就更新了 NIS server 上头的 /etc/shadow 以及 /var/yp/vbirdnis/passwd.by* 的数据库， 简单吧！一下子就同步化了。不过，如果要教育使用者使用 yppasswd 的话，他可能不太能适应， 不要紧，你可以透过修改 alias 或者是置换掉 /usr/bin/passwd 这支程序即可！那现在让我们回到 NIS 服务器端看看眞的有更动到数据库吗？

```
[root@www ~]# ll /var/yp/vbirdnis/
-rw-------. 1 root root   13836 Jul 28 13:10 netid.byname
-rw-------. 1 root root   14562 Jul 28 13:29 passwd.byname
-rw-------. 1 root root   14490 Jul 28 13:29 passwd.byuid
-rw-------. 1 root root   28950 Jul 28 13:10 protocols.byname
# 仔细看，就是那个密码档案被更动过～时间已经不一样了！再看看登录档吧！

[root@www ~]# tail /var/log/messages
Jul 28 13:29:14 www rpc.yppasswdd[1707]: update nisuser1 (uid=1001) from host
192.168.100.10 successful.
```

最终从登录档里面，我们也能够得到相关的记录！这样就非常完美啦！ ^_^

14.4 NIS 搭配 NFS 的设定在丛集计算机上的应用

刚刚在 NIS 客户端的 nisuser1 登入测试中，你应该已经发现了一件事，那就是怎么 nisuser1 没有家目录啊？这很正常啊！因为 nisuser1 的家目录是在服务器端的 /home 上头，而你在客户端登入时， 在客户端的 /home 底下根本不可能有 nisuser1 的家目录嘛！那怎办？很简单，将服务器端的 /home 挂载到客户端上面即可啊！ 那这个观念跟丛集计算机有啥关系啊？就让我们来谈谈吧！

- 什么是丛集计算机？

因为个人计算机的 CPU 速度越来越快，核心数目越来越多，因此个人计算机的效能已经不比服务器等级的大型计算机差了！ 不过，如果要用来作为计算大型数值模式的应用，即使是最快的个人计算机，还是没有办法有效的负荷的。此时你可能就得要考虑一下， 是要买超级计算机 (Top 500) 还是要自己组一部 PC 丛集计算机 (PC cluster)。

超级计算机的结构中，主要是透过内部电路将好多颗 CPU 与内存连接在一块，因为是特殊设计，因此价格非常昂贵。 如果我们可以将较便宜的个人计算机串接在一块，然后将数值运算的任务分别丢给每一部串接在一块的个人计算机， 那不就很像超级计算机了吗？没错！这就是 PC cluster 最早的想法。

但是这个作法当中有几个限制喔，因为每部计算机都需要运算相同的程序，而我们知道运算的数据都在内存当中， 而程序启动时需要给予一个身份，而程序读取的程序在每部计算机上面都需要是相同的！同时，每部计算机都需要支持平行化运算！ 所以，在 PC cluster 上面的所有计算机就得要有：

- 相同的用户帐户信息，包括账号、密码、家目录等等一大堆信息；
- 相同的文件系统，例如 /home, /var/spool/mail 以及数值程序放置的位置
- 可以搭配的平行化函式库，常见的有 MPICH, PVM...

上面的三个项目中，第一个项目我们可以透过 NIS 来处理，第二个项目则可以使用 NFS 来搞定～所以啰，你说， NIS 与 NFS 有没有可使用的空间啊？ ^_^

Tips: 由于『预测』这个玩意儿越来越重要，比如说气象预报、空气质量预报等等，而预测需要一个很庞大的模式来进行仿真的工作， 这么庞大的模拟工作需要大量的运算，在学校单位要买一部很贵的大型主机实在很不容易！不过，如果能够串接十部四核心的个人计算机的话， 那么可能只需要不到 20 万便能够组成相当于具有 40 颗 CPU 的大型主机的运算能力了！所以说，在未来 PC cluster 是一个可以发展的课题喔！



- 另一个不成材的实例

那我们有没有办法来实作一下平行化的丛集架构呢？老实说，很麻烦～不过，至少我们可以先完成前面谈到的两个组件！ 分别是 NIS 与 NFS 嘛！但是，在我们目前这个网络环境中，用户账号实在是太紊乱了～所以，如果想要将服务器的 /home 挂载到客户端的 /home，那么那个测试用的客户端可能很多本地用户都无法登入了～因此，在这个测试练习中， 我们打算这样做：

- 账号：建立大于 2000 以上的账号，账号名称为 cluser1, cluser2, cluser3 (将 cluster user 缩写为 cluser，不是少写一个 t 喔！)，且这些账号的家目录预计放置于 /rhome 目录内，以与 NIS client 本地的用户分开；
- NIS 服务器：领域名为 vbirdcluster，服务器是 www.centos.vbird (192.168.100.254)，客户端是 clientlinux.centos.vbird (192.168.100.10)；
- NFS 服务器：服务器分享了 /rhome 给 192.168.100.0/24 这个网域，且预计将所有程序放置于 /cluster 目录中。 此外，假设所有客户端都是很干净的系统，因此不需要压缩客户端 root 的身份。
- NFS 客户端：将来自 server 的文件系统都挂载到相同目录名称底下！

那就分别来实作一下啰！

- NIS 实作阶段

```
# 1\. 建立此次任务所需要的账号数据：
[root@www ~]# mkdir /rhome
[root@www ~]# useradd -u 2001 -d /rhome/cluser1 cluser1
[root@www ~]# useradd -u 2002 -d /rhome/cluser2 cluser2
[root@www ~]# useradd -u 2003 -d /rhome/cluser3 cluser3
[root@www ~]# echo password &#124; passwd --stdin cluser1
[root@www ~]# echo password &#124; passwd --stdin cluser2
[root@www ~]# echo password &#124; passwd --stdin cluser3

# 2\. 修改 NISDOMAIN 的名称
[root@www ~]# vim /etc/sysconfig/network
NISDOMAIN=vbirdcluster  &lt;==重点在改这个项目喔！
```

这个案例中，你只要做完上述的动作就即将完成了，其他的配置文件请参考前面 14.2 节所谈到的各个必要项目。 接下来当然就是重新启动 ypserv 以及制作数据库啰！

```
# 3\. 制作数据库以及重新启动所需要的服务：
[root@www ~]# nisdomainname vbirdcluster
[root@www ~]# /etc/init.d/ypserv restart
[root@www ~]# /etc/init.d/yppasswdd restart
[root@www ~]# /usr/lib64/yp/ypinit -m
```

依序一个一个指令下达！上述的这四个指令稍微有相依性关系的！所以不要错乱了顺序喔！接下来，请换到客户端进行：

1. 以 setup 进行 NIS 的设定，在领域的部分请转为 vbirdcluster 才对！
2. 做完后再以 id cluser1 确认看看。

作法太简单了，鸟哥这里就不示范啰。

- NFS 服务器的设定

```
# 1\. 设定 NFS 服务器开放的资源：
[root@www ~]# mkdir /cluster
[root@www ~]# vim /etc/exports
/rhome          192.168.100.0/24(rw,no_root_squash)
/cluster        192.168.100.0/24(rw,no_root_squash)

# 2\. 重新启动 NFS 啰：
[root@www ~]# /etc/init.d/nfs restart
[root@www ~]# showmount -e localhost
Export list for localhost:
/rhome       192.168.100.0/24
/cluster     192.168.100.0/24
```

服务器的设定是很单纯的～客户端的设定得要注意啰！

```
# 1\. 设定 NIS Client 的 mount 数据！
[root@clientlinux ~]# mkdir /rhome /cluster
[root@clientlinux ~]# mount -t nfs 192.168.100.254:/rhome   /rhome
[root@clientlinux ~]# mount -t nfs 192.168.100.254:/cluster /cluster
# 如果上述两个指令没有问题，可以将他加入 /etc/rc.d/rc.local 当中啊！

[root@clientlinux ~]# su - cluser1
[cluser1@clientlinux ~]$
```

最后你应该就能够在客户端以 cluser1 登入系统！就这么简单的将账号与文件系统同步做完啦！如果你真的想要玩一下 PC Cluster 的话，鸟哥也有写过一篇不是很成熟的 PC cluster 简易架设，有兴趣的话请自行参考：

- http://linux.vbird.org/linux_server/0600cluster.php

14.5 重点回顾

- Network Information Service (NIS) 也可以称为 Sun Yellow Pages (yp)，主要是负责在网域当中帮忙 NIS Client 端查寻账号与密码以及其他相关网络参数的服务；
- NIS server 其实就是提供本身的 /etc/passwd, /etc/shadow, /etc/group, /etc/hosts 等账号密码数据，以及相关的网络参数等，以提供网域当中 NIS Client 的搜寻之用；
- NIS 为 server/client 架构，当 NIS client 有账号登入需求时，该主机会 (1)先找自己的 /etc/passwd, (2)再前往 NIS server 搜寻相关账号资料。
- NIS 使用的软件就是 yp 这个软件，主要分为两部份， ypserv 用在 NIS Server，至于 ypbind 与 yp-tools 则用在 NIS Client 上面。
- 为加快 NIS 查询的速度，因此 NIS server 会将本机的账号数据制成传输较快的数据库档案， 并放置于 /var/yp/(nisdomainname)/ 目录当中；
- 不论是 NIS 或者是 NFS 都是藉由 RPC Server 所启用的，因此都可以使用 rpcinfo 来查寻 NIS 是否已经启动，以及该 daemon 是否已经向 portmapper (RPC server) 注册了！
- 在 NIS Server 的设定当中，最重要的一个步骤就是将账号、密码、网络参数等 ASCII 格式档案转成数据库档案 (database file)，以提供 NIS client 的查寻！而启动 ASCII 转成 database 的程序可以使用 /usr/lib64/yp/ypinit -m 或者到 /var/yp 底下执行 make 均可。
- 由于 NIS 通常使用于内部网域当中，因此 /etc/hosts 这个档案的设定相当重要！
- 若想让使用者在任一部 NIS 管辖的主机登入都可以使用同一份家目录，则需开启 NFS 提供 /home 给所有的主机挂载使用；

14.6 本章习题

- 请简单说明 NIS server 的功能与工作流程当你有多部具有相同账号的 Linux 主机时，即可利用 NIS 所提供的服务，来利用一部 NIS 主机掌控所有的 linux 主机的登入时所需查阅的账号与密码验证。流程如下：

  1. NIS Server 将自己系统内的 /etc/passwd, /etc/group, /etc/hosts 等制作成为 DBM 的数据库格式档案；
  2. NIS Client 若有用户登入的要求时，会前往 NIS Server 搜寻数据库里面的数据做为验证之用。
  3. 每次更动 NIS Server 上面的用户数据时，则 NIS Server 需要重新制作 DBM 数据库档案才行！
- 请简单说明 NIS Server/client 的架构NIS master/client 的特色为：

  1. NIS Server 的 master 先将自己的账号、密码相关档案制作成为数据库档案 (database file)；
  2. NIS Server 的 master 将自己的数据库档案传送到 slave 上面；
  3. NIS Server 的 slave 接收来自『信任的 NIS Server master 主机』的数据后，更新自己的数据库，使自己的数据库与 master 主机的数据同步；
  4. 网域当中的所有 NIS Client 查寻 NIS Server 时，会找寻『最先响应的那一部 NIS 主机的数据库内容』。

  也就是说，架设 slave NIS server 可以分担区域内 NIS 的工作！

- NIS 启动之前需要先启动那个服务，否则就无法启动成功 (提示：RPC Server)因为 NIS 是 RPC Server 的一种，所以必须要启动 rpcbind 这个 daemon 才行！
- 我的 NIS 域名为 bird ，另外，我主机的 IP 与主机名为 192.168.5.1/bird.nis.org ，请问要这些信息需要设定在 NIS Server 的哪些档案之内？域名可以直接手动下达『 nisdomainname bird 』也可以写入 /etc/sysconfig/network 里面『NISDOMAIN=bird』；IP 与 主机名 需要写入在 /etc/hosts 里面。
- /etc/nsswitch.conf 的功能为何？如果我想要让密码查寻先本地的密码文件，再查寻 NIS ，需要如何设定？该档案的功能很多，在 DNS 方面，可以用来决定正、反解的顺序，至于密码则可以用来判断何者为先！如果需要先查本机再查 NIS 的密码时，需要的参数：passwd: files nis shadow: files nis
- 如果我想要增加网域当中一个新的账号：newaccount，并且这个 newaccount 可以让 NIS Client 查寻到他的账号与密码，需要进行哪些步骤？

  1. 先登入 NIS Server 以 useradd newaccount 以及 passwd newaccount 来新增账号；
  2. 制作密码数据库：『/usr/lib64/yp/ypinit -m』
  3. 重新启动：『/etc/rc.d/init.d/ypserv restart ; /etc/rc.d/init.d/yppasswdd restart』。

- 实作范例题：底下是我的网域参数特征： network/netmask:192.168.1.0/255.255.255.0 NIS server : 192.168.1.100 (hostname: server.nis.test) NIS cient: 192.168.1.200 (hostname: client1.nis.test) NIS domain name: nis.test 利用上面的参数来设定 NIS 架构，请一步一步的写下你的设定。请自行参考本章节的内容设定

14.7 参考数据与延伸阅读

- Study Area 之 NIS 服务器架设：http://www.study-area.org/linux/servers/linux_nfs.htm
- NIS 官方网站：http://www.linux-nis.org/
- NIS HOW-TO：http://www.linux-nis.org/nis-howto/HOWTO/index.html

---

2003/05/06：第一次完成日期！ 2003/09/16：稍微加入一些信息与微幅修改版面！ 2006/09/22：将旧的文章移动到 此处 2006/10/11：啊！过了好久了！修改过程当中历经搬家，所以文章产生较慢啊！这次多加入 NIS slave server 说！ 2011/03/13：将旧的基于 CentOS 4.x 的版本移动到 此处 2011/03/16：因为 NIS 快要被 LDAP 取代，所以将比较复杂的 slave 部分删去了～ 2011/07/28：将基于 CentOS 5.x 的版本移动到此处 加入 /etc/idmapd.conf 的相关 domain 功能！要讲要讲！重要重要！

第十五章、时间服务器：NTP 服务器

最近更新日期：2011/07/29

计算机内部所记录的时钟是记载于 BIOS (CMOS) 内的，但如果你的计算机上面的电池没电了，或者是某些特殊因素导致 BIOS 数据被清除， 此时计算机的时间就会不准。同时，某些操作系统程序的问题，也可能导致我们看到的时间与现实社会不相同的情况。 所以我们都会调整一下时间，好让计算机系统的时间可以一直保持正确的状态。 在实际生活中，我们可以透过电视台、广播电台、电话等等来调整我们的手表，那么如果是在网络上呢？ 该如何让我们的主机随时保持正确的时间信息？这就需要 NTP 这个服务器啰。

15.1 关于时区与网络校时的通讯协议

时间对于现代人来说是很重要的，因为『 Time is money 』。既然时间如此重要，对于因特网来说应该也是很重要吧？ 为什么呢？还记得我们在基础学习篇第三版第十九章、登录档分析吧？ 如果你架设了一个登录档服务器的话，那么总得要分析每个主机所传来的登录文件信息吧？如果每一部主机的时间都不相同， 那如何判断问题发生的时间点？所以啰，『每一部主机的时间同步化』就很重要了。

每一部主机时间同步化的重要性当然不只如此，包括之前谈到的 DHCP 客户端/服务器端所需要的租约时间限制、 网络侦测时所需要注意的时间点、刚刚谈到的登录文件分析功能、具有相关性的主机彼此之间的错误侦测、前一章谈到的丛集计算机群等等， 都需要具有相同的时间才能够捉出问题呢。好了，底下咱们就来聊一聊，如何利用网络来进行主机的时间同步化吧！

15.1.1 什么是时区？全球有多少时区？GMT 在那个时区？

因为地球是圆的，所以同一个时刻，在地球的一边是白天，一边是黑夜。而因为人类使用一天 24 小时的制度，所以，在地球对角的两边就应该差了 12 个小时才对。由于同一个时间点上面， 整个地球表面的时间应该都不一样，为了解决这个问题，所以可以想见的，地球就被分成 24 个时区了！

那么这 24 个时区是依据什么来划分的呢？由于地球被人类以『经纬度』坐标来进行定位，而经度为零的地点在英国『格林威治』这个城市所在的纵剖面上， (注：所谓的纵剖面就是由南极切到北极的直线，而横切面就是与赤道平行的切线)，如下图所示：

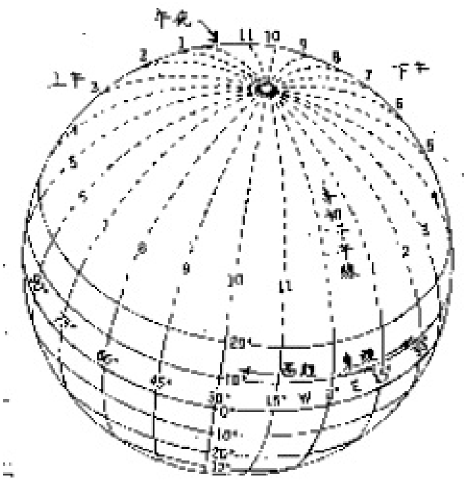


图 15.1-1、地球的子午线、经纬度与时区的分隔概念

因为绕地球一圈是 360 度角，这 360 度角共分为 24 个时区，当然一个时区就是 15 度角啦！又由于是以格林威治时间为标准时间 (Greenwich Mean Time, GMT 时间)，加上地球自转的关系，因此，在格林威治以东的区域时间是比较快的(+小时)，而以西的地方当然就是较慢

啰！

以台湾为例，因为台湾所在地约为东经 120 度北纬 25 度左右，又因为台湾在格林威治的东方 (废话！因为是东经嘛！ ^_^)，因此台湾本地时间 (local time) 会比 GMT 时间快 8 小时 (GMT + 8)。当格林威治时间为零点，台湾就已经是早上八点了！底下约略列出各个时区的名称与所在经度，以及与 GMT 时间的时差：

标准时区	经度	时差
GMT , Greenwich Mean Time	0 W/E	标准时间
CET , Central European	15 E	+1 东一区
EET , Eastern European	30 E	+2 东二区
BT , Baghdad	45 E	+3 东三区
USSR, Zone 3	60 E	+4 东四区
USSR, Zone 4	75 E	+5 东五区
Indian, First	82.3E	+5.5东五半区
USSR, Zone 5	90 E	+6 东六区
SST , South Sumatra	105 E	+7 东七区
JT , Java	112 E	+7.5东七半区
CCT , China Coast (台湾所在地)	120 E	+8 东八区
JST , Japan	135 E	+9 东九区
SAST, South Australia	142 E	+9.5东九半区
GST , Guam	150 E	+10 东十区
NZT , New Zealand	180 E	+12 东十二区
Int'l Date Line	180 E/W	国际日期变更线
BST , Bering	165 W	-11 西十一区
SHST, Alaska/Hawaiian	150 W	-10 西十区
YST , Yukon	135 W	-9 西九区
PST , Pacific	120 W	-8 西八区
MST , Mountain	105 W	-7 西七区
CST , Central	90 W	-6 西六区
EST , Eastern	75 W	-5 西五区
AST , Atlantic	60 W	-4 西四区
Brazil, Zone 2	45 W	-3 西三区
AT , Azores	30 W	-2 西二区
WAT , West Africa	15 W	-1 西一区

所以啰，台湾时间是 GMT + 8 就很容易推算出来了吧！要特别留意的是，很多朋友在安装 Linux 的时候， 总是会发现目前的时间慢或者快了 8 小时，不要怀疑，绝对与时区有关！ 赶紧给他查一下如何调整时区吧！ ^_^。

另外，在上表中有个比较有趣的时区，那就是在太平洋上面的国际日期变更线了！我们刚刚说，在格林威治的东边时间会较快， 而在西边时间会较慢，但是两边各走了 180 度之后就会碰头啊！那不就刚好差了 24 小时吗？没错啦！ 所以才订定为『国际日期变更线』啊！国际日期变更线刚好在太平洋上面，因此，如果你有坐飞机到美国的经验应该会发现，咦！ 怎么出发的时间是星期六下午，坐了 13 个小时的飞机到了美国还是星期六！因为刚好通过了国际日期变更线， 日期减少了一天喔！如果反过来，由美国到台湾，日期就会多加一天喔！ ^_^

15.1.2 什么是夏季节约时间 (daylight savings)？

时区的概念先建立起来之后，现在再来谈一谈，那么什么是『夏季节约时间 (或称日光节约时间)』？既然是『夏季节约时间』当然主要是与夏天有关啦！ 因为地球在运行的时候是呈现一个倾斜角在绕太阳运转的，所以才有春夏秋冬 (这个大家应该都知道啦)，在夏天的时候，白天的时间会比较长，所以为了节约用电， 因此在夏天的时候某些地区会将他们的时间定早一小时，也就是说，原本时区是 8 点好了， 但是因为夏天太阳比较早出现，因此把时间向前挪，在原本 8 点的时候，订定为该天的 9 点 (时间提早一小时)～如此一来， 我们就可以利用阳光照明，省去了花费电力的时间，因此才会称之为夏季节约时间！

因为台湾实在是太小了，并没有横跨两个时区，因此，夏季节约时间对我们来说，虽然还是有帮助啦！ 不过，似乎没有特别推行的样子说～

15.1.3 Coordinated Universal Time (UTC)与系统时间的误差

了解了一些时区的概念之后，这里要谈的是『什么是正确的时间』。 在 1880 年代的时间标准是以 GMT 时间为主的，但是 GMT 时间是以太阳通过格林威治的那一刻来作为计时的标准。 然而我们都知道啊，地球自转的轨道以及公转的轨道并非正圆，加上地球的自转速度好像有逐年递减的问题， 所以这个 GMT 时间与我们目前计时的时间就有点不一样了。(注1)

在计算时间的时候，最准确的计算应该是使用『原子震荡周期』所计算的物理时钟了 (Atomic Clock, 也被称为原子钟)，这也被定义为标准时间 (International Atomic Time)。而我们常常看见的 UTC 也就是 Coordinated Universal Time (协和标准时间)就是利用这种 Atomic Clock 为基准所定义出来的正确时间。例如 1999 年在美国启用的原子钟 NIST F-1， 他所产生的时间误差每两千年才差一秒钟！真的是很准呐！这个 UTC 标准时间虽然与 GMT 时间放在同一个时区为基准， 不过由于计时的方式不同，UTC 时间与 GMT 时间有差不多 16 分钟的误差呢！(注2)

事实上，在我们的身边就有很多的原子钟，例如石英表，还有计算机主机上面的 BIOS 内部就含有一个原子钟在纪录与计算时间的进行呐！不过由于原子钟主要是利用计算芯片 (crystal) 的原子震荡周期去计时的，这是因为每种芯片都有自己的独特的震荡周期之故。 然而因为这

种芯片的震荡周期在不同的芯片之间多多少少都会有点差异性，甚至同一批芯片也可能会或多或少有些许的差异 (就连温度也可能造成这样的误差呢)，因此也就造成了 BIOS 的时间会经常的给他快了几秒或者慢了几秒。

或许你会认为，BIOS 定时器每天快个五秒也没有什么了不起的，不过如果你再仔细的算一算，会发现，一天快五秒，那么一个月快 2.5 分钟，一年就快了 75 分钟了！所以说，呵呵！时间差是真的会存在的！那么如果你的计算机真的有这样的情况， 那要怎么来重新校正时间呢？那就需要『网络校时』(Network Time Protocol, NTP) 的功能了！底下我们就谈一谈那个 NTP 的 daemon 吧！

15.1.4 NTP 通讯协议

老实说， Linux 操作系统的计时方式主要是由 1970/01/01 开始计算总秒数，因此，如果你还记得 date 这个指令的话， 会发现它有个 +%s 的参数，可以取得总秒数，这个就是软件时钟。但，如同前面说的，计算机硬件主要是以 BIOS 内部的时间为主要的时间依据 (硬件时钟)，而偏偏这个时间可能因为 BIOS 内部芯片本身的问题，而导致 BIOS 时间与标准时间 (UTC) 有一点点的差异存在！所以为了避免主机时间因为长期运作下所导致的时间偏差，进行时间同步 (synchronize) 的工作就显的很重要了！

- 软件时钟：由 Linux 操作系统根据 1970/01/01 开始计算的总秒数；
- 硬件时钟：主机硬件系统上面的时钟，例如 BIOS 记录的时间；

那么怎么让时间同步化呢？想一想，如果我们选择几部主要主机 (Primary server) 调校时间，让这些 Primary Servers 的时间同步之后，再开放网络服务来让 Client 端联机，并且提供 Client 端调整自己的时间，不就可以达到全部的计算机时间同步化的运作了吗！那么什么协议可以达到这样的功能呢？那就是 Network Time Protocol ，另外还有 Digital Time Synchronization Protocol (DTSS) 也可以达到相同的功能！ 不过，到底 NTP 这个 daemon 是如何让 Server 与 Client 同步他们的时间呢？

1. 首先，主机当然需要启动这个 daemon ，之后，
2. Client 会向 NTP Server 发送出调校时间的 message ，
3. 然后 NTP Server 会送出目前的标准时间给 Client ，
4. Client 接收了来自 Server 的时间后，会据以调整自己的时间，就达成了网络校时咯！

在上面的步骤中你有没有想到一件事啊，那就是如果 Client 到 Server 的讯息传送时间过长怎么办？举例来说，我在台湾以 ADSL 的 PC 主机，联机到美国的 NTP Server 主机进行时间同步化要求，而美国 NTP Server 收到我的要求之后，就发送当时的正确时间给我，不过， 由美国将数据传送回我的 PC 时，时间可能已经延迟了 10 秒钟去了！这样一来，我的 PC 校正的时间是 10 秒钟前的标准时间喔！此外，如果美国那么 NTP 主机有太多的人喜欢上去进行网络校时了，所以 loading (负荷) 太重啦！导致讯息的回传又延迟的更为严重！那怎么办？

为了这些延迟的问题，有一些 program 已经开发了自动计算时间传送过程的误差，以更准确的校准自己的时间！当然啦，在 daemon 的部分，也同时以 server/client 及 master/slave 的架构来提供用户进行网络校时的动作！所谓的 master/slave 就有点类似 DNS 的系统咯！举例来说，台湾的标准时间主机去国际标准时间的主机校时， 然后各大专院校再到台湾的标准时间校时，然后我们再到各大专院校的标准时间校时！这样一来，那几部国际标准时间主机 (Time server) 的 loading 就不至于太大，而我们也可以很快速的达到正确的网络校时的目的呢！台湾常见的 Time Server 有 (注3)：

- tick.stdtime.gov.tw
- tock.stdtime.gov.tw
- time.stdtime.gov.tw
- clock.stdtime.gov.tw
- watch.stdtime.gov.tw

至于 ntp 这个 daemon 是以 port 123 为连结的埠口 (使用 UDP 封包)，所以我们要利用 Time server 来进行时间的同步更新时，就得要使用 NTP 软件提供的 ntpdate 来进行 port 123 的联机喔！关于网络校时更多的说明，可以到 NTP 的官方网站 (注4) 上察看喔！

15.1.5 NTP 服务器的阶层概念

如前所述，由于 NTP 时间服务器采用类似阶层架构 (stratum) 来处理时间的同步化， 所以他使用的是类似一般 server/client 的主从架构。网络社会上面有提供一些主要与次要的时间服务器， 这些均属于第一阶及第二阶的时间服务器 (stratum-1, stratum-2) ，如下所示：

- 主要时间服务器： http://support.ntp.org/bin/view/Servers/StratumOneTimeServers
- 次要时间服务器： http://support.ntp.org/bin/view/Servers/StratumTwoTimeServers

由于这些时间服务器大多在国外，所以我们是否要使用这些服务器来同步化自己的时间呢？其实如果台湾地区已经有标准时间服务器的话，用那部即可，不需要联机到国外啦！浪费带宽与时间啊！ 而如前面提到的，台湾地区已经有标准的时间服务器了，所以当然我们可以直接选择台湾地区的 NTP 主机即可。

如果你评估一下，确定有架设 NTP 的需求时，我们可以直接选择台湾地区的上层 NTP 来同步化时间即可。 举例来说 tock.stdtime.gov.tw 这个国家单位的主机应该是比较适合的。一般来说，我们在进行 NTP 主机的设定时，都会先选择多部上层的 Time Server 来做为我们这一部 NTP Server 的校正之用，选择多部的原因是因为可以避免因为某部时间服务器突然挂点时， 其他主机仍然可以提供我们的 NTP 主机来自我更新啊！然后我们的 NTP Server 才提供给自己的 Client 端更新时间。如此一来，国家单位的 tock.stdtime.gov.tw 负载才不会太大，而我们的 Client 也可以很快速的达到校时的动作！

Tips: 其实 NTP 的阶层概念与 DNS 很类似啦，当你架设一部 NTP 主机，这部 NTP 所向上要求同步化的那部主要主机为 stratum-1 时，那么你的 NTP 就是 stratum-2 啰！举例来说，如果我们的 NTP 是向台湾的 tock.stdtime.gov.tw 这部 stratum-2 的主机要求时间同步化，那我们的主机即为 stratum-3 ，如果还有其他的 NTP 主机向我们要求时间同步， 那么该部主机则会是 stratum-4 啦！就这样啊～ 那最多可以有几个阶层？最多可达 15 个阶层喔！



15.2 NTP 服务器的安装与设定

NTP 服务器也是一个很容易就可以架设成功的玩意儿，不过这个软件在不同的 distribution 上面可能有不一样的名称， 你要作的其实就是将他安装起来之后，规定一部上层 NTP 服务器来同步化你的时间即可啊！ 如果你只是想要进行你自己单部主机的时间同步化，别架设 NTP ，直接使用 NTP 客户端软件即可喔！

15.2.1 所需软件与软件结构

在 CentOS 6.x 上头，你所需要的软件其实仅有 ntp 这个玩意儿而已，请自行使用 rpm 去找找看，若没有安装，请利用 yum install ntp 即可啊！不过，我们还需要时区相关的数据文件，所以你需要的软件有：

- ntp： 就是 NTP 服务器的主要软件啦，包括配置文件以及执行档等等。
- tzdata： 软件名称为『 Time Zone data 』的缩写，提供各时区对应的显示格式。

与时间及 NTP 服务器设定相关的配置文件与重要数据文件有底下几个：

- /etc/ntp.conf： 就是 NTP 服务器的主要配置文件，也是唯一的一个；
- /usr/share/zoneinfo/： 由 tzdata 所提供，为各时区的时间格式对应档。 例如台湾地区的时区格式对应档案在 /usr/share/zoneinfo/Asia/Taipei 就是了！这个目录里面的档案与底下要谈的两个档案 (clock 与 localtime) 是有关系的喔！
- /etc/sysconfig/clock： 设定时区与是否使用 UTC 时间钟的配置文件。 每次开机后 Linux 会自动的读取这个档案来设定自己系统所默认要显示的时间说！举个例子来说， 在我们台湾地区的本地时间设定中，这个档案内应该会出现一行『ZONE="Asia/Taipei"』的字样， 这表示我们的时间配置文件案『要取用 /usr/share/zoneinfo/Asia/Taipei 那个档案』的意思！
- /etc/localtime： 这个档案就是『本地端的时间配置文件』啦！刚刚那个 clock 档案里面规定了使用的时间配置文件 (ZONE) 为 /usr/share/zoneinfo/Asia/Taipei ，所以说这就是本地端的时间了，此时 Linux 系统就会将 Taipei 那个档案复制一份成为 /etc/localtime ，所以未来我们的时间显示就会以 Taipei 那个时间配置文件案为准。

至于在常用于时间服务器与修改时间的指令方面，主要有底下这几个啦：

- /bin/date： 用于 Linux 时间 (软件时钟) 的修改与显示的指令；
- /sbin/hwclock： 用于 BIOS 时钟 (硬件时钟) 的修改与显示的指令。 这是一个 root 才能执行的指令，因为 Linux 系统上面 BIOS 时间与 Linux 系统时间是分开的，所以使用 date 这个指令调整了时间之后，还需要使用 hwclock 才能将修改过后的时间写入 BIOS 当

中！

- /usr/sbin/ntpd： 主要提供 NTP 服务的程序啰！配置文件为 /etc/ntp.conf
- /usr/sbin/ntpdate： 用于客户端的时间校正，如果你没有要启用 NTP 而仅想要使用 NTP Client 功能的话，那么只会用到这个指令而已啦！

例题：假设你的笔记本电脑安装 CentOS 这套系统，而且选择的时区为台湾。现在，你将有一个月的时间要出差到美国的纽约去， 你会带着这个笔电，那么到了美国之后，时间会不一致啊！你该如何手动的调整时间参数呢？答：因为时区数据文件在 /usr/share/zoneinfo 内，在该目录内会找到 /usr/share/zoneinfo/America/New_York 这个时区档。 而时区配置文件在 /etc/sysconfig/clock ，且目前的时间格式在 /etc/localtime ，所以你应该这样做：

```
[root@www ~]# date
Thu Jul 28 15:08:39 CST 2011  &lt;==重点是 CST 这个时区喔！

[root@www ~]# vim /etc/sysconfig/clock
ZONE="America/New_York"       &lt;==改的是这里啦！

[root@www ~]# cp /usr/share/zoneinfo/America/New_York /etc/localtime
[root@www ~]# date
Thu Jul 28 03:09:21 EDT 2011  &lt;==时区与时间都改变了！
```

这个范例做完之后，记得将这两个档案改回来！不然以后你的时间都是美国时间啦！

接下来，我们先来谈一谈如何设计那个 /etc/ntp.conf 吧！

15.2.2 主要配置文件 ntp.conf 的处理

由于 NTP 服务器的设定需要有上游服务器的支持才行，因此请回头参考一下 15.1.4 及 15.1.5 的介绍，这样才能够理解为何底下的设定是这样呦！好了，我假设俺的 NTP 服务器所需要设定的架构如下：

- 我的上层 NTP 服务器共有 tock.stdtime.gov.tw, tick.stdtime.gov.tw, time.stdtime.gov.tw 三部，其中以 tock.stdtime.gov.tw 最优先使用 (prefer)；
- 不对 Internet 提供服务，仅允许来自内部网域 192.168.100.0/24 的查询而已；
- 侦测一些 BIOS 时钟与 Linux 系统时间的差异并写入 /var/lib/ntp/drift 档案当中。

好了，先让我们谈一谈如何在 ntp.conf 里面设定权限控制吧！

- 利用 restrict 来管理权限控制

在 ntp.conf 档案内可以利用『 restrict 』来控管权限，这个参数的设定方式为：

```
restrict [你的IP] mask [netmask_IP] [parameter]
```

其中 parameter 的参数主要有底下这些：

- ignore： 拒绝所有类型的 NTP 联机；
- nomodify： 客户端不能使用 ntpc 与 ntpq 这两支程序来修改服务器的时间参数， 但客户端仍可透过这部主机来进行网络校时的；
- noquery： 客户端不能够使用 ntpq, ntpc 等指令来查询时间服务器，等于不提供 NTP 的网络校时啰；
- notrap： 不提供 trap 这个远程事件登录 (remote event logging) 的功能。
- notrust： 拒绝没有认证的客户端。

那如果你没有在 parameter 的地方加上任何参数的话，这表示『该 IP 或网段不受任何限制』的意思喔！一般来说，我们可以先关闭 NTP 的权限，然后再一个一个的启用允许登入的网段。

- 利用 server 设定上层 NTP 服务器

上层 NTP 服务器的设定方式为：

```
server [IP or hostname] [prefer]
```

在 server 后端可以接 IP 或主机名，鸟哥个人比较喜欢使用 IP 来设定说！至于那个 perfer 表示『优先使用』的服务器啰～有够简单吧！

- 以 driftfile 记录时间差异

设定的方式如下：

```
driftfile [可以被 ntpd 写入的目录与档案]
```

因为预设的 NTP Server 本身的时间计算是依据 BIOS 的芯片震荡周期频率来计算的，但是这个数值与上层 Time Server 不见得会一致啊！所以 NTP 这个 daemon (ntpd) 会自动的去计算我们自己主机的频率与上层 Time server 的频率，并且将两个频率的误差记录下来，记录下来的档案就是在 driftfile 后面接的完整档名当中了！关于档名你必须要知道：

- driftfile 后面接的档案需要使用完整路径文件名；
- 该档案不能是连结档；
- 该档案需要设定成 ntpd 这个 daemon 可以写入的权限。
- 该档案所记录的数值单位为：百万分之一秒 (ppm)。

driftfile 后面接的档案会被 ntpd 自动更新，所以他的权限一定要能够让 ntpd 写入才行。在 CentOS 6.x 预设的 NTP 服务器中，使用的 ntpd 的 owner 是 ntp ，这部份可以查阅 /etc/sysconfig/ntpd 就可以知道啦！

- keys [key_file]

除了以 restrict 来限制客户端的联机之外，我们也可以透过密钥系统来给客户端认证， 如此一来可以让主机端更放心了。不过在这个章节里面我们暂不讨论这个部分，有兴趣的朋友可以参考 ntp-keygen 这个指令的相关说明喔！

根据上面的说明，我们最终可以取得这样的配置文件案内容喔 (底下仅修改部分数据，保留大部分的设定值喔)！

```
[root@www ~]# vim /etc/ntp.conf
# 1\. 先处理权限方面的问题，包括放行上层服务器以及开放区网用户来源：
restrict default kod nomodify notrap nopeer noquery     &lt;==拒绝 IPv4 的用户
restrict -6 default kod nomodify notrap nopeer noquery  &lt;==拒绝 IPv6 的用户
restrict 220.130.158.71   &lt;==放行 tock.stdtime.gov.tw 进入本 NTP 服务器
restrict 59.124.196.83    &lt;==放行 tick.stdtime.gov.tw 进入本 NTP 服务器
restrict 59.124.196.84    &lt;==放行 time.stdtime.gov.tw 进入本 NTP 服务器
restrict 127.0.0.1        &lt;==底下两个是默认值，放行本机来源
restrict -6 ::1
restrict 192.168.100.0 mask 255.255.255.0 nomodify &lt;==放行区网来源

# 2\. 设定主机来源，请先将原本的 [0&#124;1&#124;2].centos.pool.ntp.org 的设定批注掉：
server 220.130.158.71 prefer  &lt;==以这部主机为最优先
server 59.124.196.83
server 59.124.196.84

# 3.预设时间差异分析档案与暂不用到的 keys 等，不需要更动它：
driftfile /var/lib/ntp/drift
keys      /etc/ntp/keys
```

这样就设定妥当了，准备来启动 NTP 服务吧！

15.2.3 NTP 的启动与观察

设定完 ntp.conf 之后就可以启动 ntp 服务器了。启动与观察的方式如下：

```
# 1\. 启动 NTP
[root@www ~]# /etc/init.d/ntpd start
[root@www ~]# chkconfig ntpd on
[root@www ~]# tail /var/log/messages  &lt;==自行检查看看有无错误

# 2\. 观察启动的埠口看看：
[root@www ~]# netstat -tlunp &#124; grep ntp
Proto Recv-Q Send-Q Local Address       Foreign Address  PID/Program name
udp        0      0 192.168.100.254:123 0.0.0.0:*        3492/ntpd
udp        0      0 192.168.1.100:123   0.0.0.0:*        3492/ntpd
udp        0      0 127.0.0.1:123       0.0.0.0:*        3492/ntpd
udp        0      0 0.0.0.0:123         0.0.0.0:*        3492/ntpd
udp        0      0 ::1:123             :::*             3492/ntpd
udp        0      0 :::123              :::*             3492/ntpd
# 主要是 UDP 封包，且在 port 123 这个埠口的啦！
```

这样就表示我们的 NTP 服务器已经启动了，不过要与上层 NTP 服务器联机则还需要一些时间， 通常启动 NTP 后约在 15 分钟内才会和上层 NTP 服务器顺利连接上。 那要如何确认我们的 NTP 服务器有顺利的更新自己的时间呢？你可以使用底下几个指令来查阅喔 (请自行等

待数分钟后再以下列指令查阅)：

```
[root@www ~]# ntpstat
synchronised to NTP server (220.130.158.71) at stratum 3
   time correct to within 538 ms
   polling server every 128 s
```

这个指令可以列出我们的 NTP 服务器有跟上层联机否。由上述的输出结果可以知道，时间有校正约 538 * 10^(-3) 秒，且每隔 64 秒会主动去更新时间喔！

```
[root@www ~]# ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*tock.stdtime.go 59.124.196.87    2 u   19  128  377   12.092   -0.953   0.942
+59-124-196-83.H 59.124.196.86    2 u    8  128  377   14.154    7.616   1.533
+59-124-196-84.H 59.124.196.86    2 u    2  128  377   14.524    4.354   1.079
```

这个 ntpq -p 可以列出目前我们的 NTP 与相关的上层 NTP 的状态，上头的几个字段的意义为：

- remote：亦即是 NTP 主机的 IP 或主机名啰～注意最左边的符号
  - 如果有『 * 』代表目前正在作用当中的上层 NTP
  - 如果是『 + 』代表也有连上线，而且可作为下一个提供时间更新的候选者。
- refid：参考的上一层 NTP 主机的地址
- st：就是 stratum 阶层啰！
- when：几秒钟前曾经做过时间同步化更新的动作；
- poll：下一次更新在几秒钟之后；
- reach：已经向上层 NTP 服务器要求更新的次数
- delay：网络传输过程当中延迟的时间，单位为 10^(-6) 秒
- offset：时间补偿的结果，单位与 10^(-3) 秒
- jitter：Linux 系统时间与 BIOS 硬件时间的差异时间， 单位为 10^(-6) 秒。

事实上这个输出的结果告诉我们，时间真的很准了啦！因为差异都在 0.001 秒以内， 可以符合我们的一般使用了。另外，你也可以检查一下你的 BIOS 时间与 Linux 系统时间的差异，就是 /var/lib/ntp/drift 这个档案的内容，就能了解到咱们的 Linux 系统时间与 BIOS 硬件时钟到底差多久？单位为 10^(-6) 秒啦！

要让你的 NTP Server/Client 真的能运作，在上述的动作中得注意：

- 上述的 ntpstat 以及 ntpq -p 的输出结果中，你的 NTP 服务器真的要能够连结上层 NTP 才行喔！ 否则你的客户端将无法对你的 NTP 服务器进行同步更新的！重要重要！
- 你的 NTP 服务器时间不可与上层差异太多。举例来说，鸟哥测试 NTP 服务器约在 2011/7/28 下午， 如果我的服务器时间原本是错误的 2010/7/28，足足差了一年，那么上层服务器恐怕就不会将正确的时间传给我！ 这时就会造成困扰了！
- 服务器防火墙在 UDP port 123 有没有开啊？要特别注意的呢！

- 等待的时间够不够长？鸟哥设定 NTP 等过最久的时间大约是一小时！你有等这么久过否？

15.2.4 安全性设定

NTP 服务器在安全的相关性方面，其实刚刚我们在 /etc/ntp.conf 里面的 restrict 参数中就已经设定了 NTP 这个 daemon 的服务限制范围了！不过，在防火墙 iptables 的部分，还是需要开启联机监听的啦！所以，在你的 iptables 规则的 scripts 当中，需要加入这一段 (我是以开放 192.168.100.0/24 这个网域作为范例的！)

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.allow
iptables -A INPUT -i $EXTIF -p udp -s 192.168.100.0/24 --dport 123 -j ACCEPT

[root@www ~]# /usr/local/virus/iptables/iptables.rule
```

若还要开放其他的网段或者客户端主机，请自行修改 /etc/ntpd.conf 以及你的防火墙机制咯！

15.3 客户端的时间更新方式

上头介绍了 NTP 服务器的安装与设定，如果我们仅有十部不到的主机时，老实说，实在没有架设 NTP 服务器的需求。 只要能够在你的主机上头以 NTP 客户端软件来进行网络校时就能够同步化时间了，没必要时时刻刻进行时间的校正吧！^_^！ 但是，如果是类似一定要时间同步的丛集计算机群或登录服务器群，那就得要使用时间服务器比较好啰！

15.3.1 Linux 手动校时工作： date, hwclock

先来复习一下前面谈到的重点，那就是 Linux 操作系统当中其实有两个时间，分别是：

- 软件时钟： Linux 自己的系统时间，由 1970/01/01 开始记录的时间参数
- 硬件时钟： 计算机系统在 BIOS 记录的实际时间，这也是硬件所记录的

在软件时钟方面，我们可以透过 date 这个指令来进行手动修订，但如果要修改 BIOS 记录的时间，就得要使用 hwclock 这个指令来写入才行。相关的用法如下：

```
[root@clientlinux ~]# date MMDDhhmmYYYY
选项与参数：
MM：月份
DD：日期
hh：小时
mm：分钟
YYYY：公元年

# 1\. 修改时间成为 1 小时后的时间该如何是好？
[root@clientlinux ~]# date
Thu Jul 28 15:33:38 CST 2011

[root@clientlinux ~]# date 072816332011
Thu Jul 28 16:33:00 CST 2011
# 瞧！时间立刻就变成一个小时后了！
```

```
[root@clientlinux ~]# hwclock [-rw]
选项与参数：
-r ：亦即 read ，读出目前 BIOS 内的时间参数；
-w ：亦即 write ，将目前的 Linux 系统时间写入 BIOS 当中啊！

# 2\. 查阅 BIOS 时间，并且写入更改过的时间啰！
[root@clientlinux ~]# date; hwclock -r
Thu Jul 28 16:34:00 CST 2011
Thu 28 Jul 2011 03:34:57 PM CST  -0.317679 seconds
# 看一看，是否刚好差异约一个小时啊！这就是 BIOS 时间！

[root@clientlinux ~]# hwclock -w; hwclock -r; date
Thu 28 Jul 2011 04:35:12 PM CST  -0.265656 seconds
Thu Jul 28 16:35:11 CST 2011
# 这样就写入啰～所以软件时钟与硬件时钟就同步啦！很简单吧！
```

这样可以了解了吗？当我们进行完 Linux 时间的校时后，还需要以 hwclock 来更新 BIOS 的时间，因为每次重新启动的时候，系统会重新由 BIOS 将时间读出来，所以， BIOS 才是重要的时间依据呐。

15.3.2 Linux 的网络校时

在 Linux 的环境当中可利用 NTP 的客户端程序，亦即是 ntpdate 这支程序就能够进行时间的同步化。 不过你要知道的是，因为 NTP 服务器本来就会与上层时间服务器进行时间的同步化， 所以在预设的情况下，NTP 服务器不可以使用 ntpdate ！也就是说 ntpdate 与 ntpd 不能同时启用的。 所以你不要在 NTP server 上头执行这个指令呦！我们就来看看如何处理吧！

```
[root@clientlinux ~]# ntpdate [-dv] [NTP IP/hostname]
选项与参数：
-d ：进入除错模式 (debug) ，可以显示出更多的有效信息。
-v ：有较多讯息的显示。

[root@clientlinux ~]# ntpdate 192.168.100.254
28 Jul 17:19:33 ntpdate[3432]: step time server 192.168.100.254 offset -2428.396146 sec
# 最后面会显示微调的时间有多少 (offset)，因为鸟哥这部主机时间差很多，所以秒数...

[root@clientlinux ~]# date; hwclock -r
四  7月 28 17:20:27 CST 2011
公元2011年07月28日 (周四) 18时19分26秒  -0.752303 seconds
# 知道鸟哥想要表达什么吗？对啊！还得 hwclock -w 写入 BIOS 时间才行啊！

[root@clientlinux ~]# vim /etc/crontab
# 加入这一行去！
10 5 * * * root (/usr/sbin/ntpdate tock.stdtime.gov.tw && /sbin/hwclock -w) &&gt; /dev/nu
```

使用 crontab 之后，每天 5:10 Linux 系统就会自动的进行网络校时啰！相当的简易吧！不过，这个方式仅适合不要启动 NTP 的情况。如果你的机器数量太多了，那么客户端最好也启动一下 NTP 服务！透过 NTP 去主动的更新时间吧！ 如何达成这个动作呢？也很简单啊，修改 /etc/ntp.conf 即可：

```
[root@clientlinux ~]# ntpdate 192.168.100.254
# 由于 ntpd 的 server/client 之间的时间误差不允许超过 1000 秒，
# 因此你得先手动进行时间同步，然后再设定与启动时间服务器呦！

[root@clientlinux ~]# vim /etc/ntp.conf
#server 0.centos.pool.ntp.org
#server 1.centos.pool.ntp.org
#server 2.centos.pool.ntp.org
restrict 192.168.100.254  &lt;==放行服务器来源！
server 192.168.100.254    &lt;==这就是服务器！
# 很简单，就是将原本的 server 项目批注，加入我们要的服务器即可

[root@clientlinux ~]# /etc/init.d/ntpd start
[root@clientlinux ~]# chkconfig ntpd on
```

然后取消掉 crontab 的更新程序，这样你的 client 计算机就会主动的到 NTP 服务器去更新啰！也是轻松愉快啊！ 不过针对客户端来说，鸟哥还是比较习惯使用 crontab 的方式来处理就是了。

15.3.3 Windows 的网络校时

或许你一直都没发现，其实 Windows 在预设的情况当中，已经帮我们处理了网络校时的工作喔！不管你愿不愿意... 你可以将鼠标的指针指在任务栏右下角的时间以如下的方式来查阅一下网络时间服务器的设定：



图 15.3-1、Windows 7 提供的网络校时功能

点选上图中的『变更日期与时间设定值』，出现如下图示：

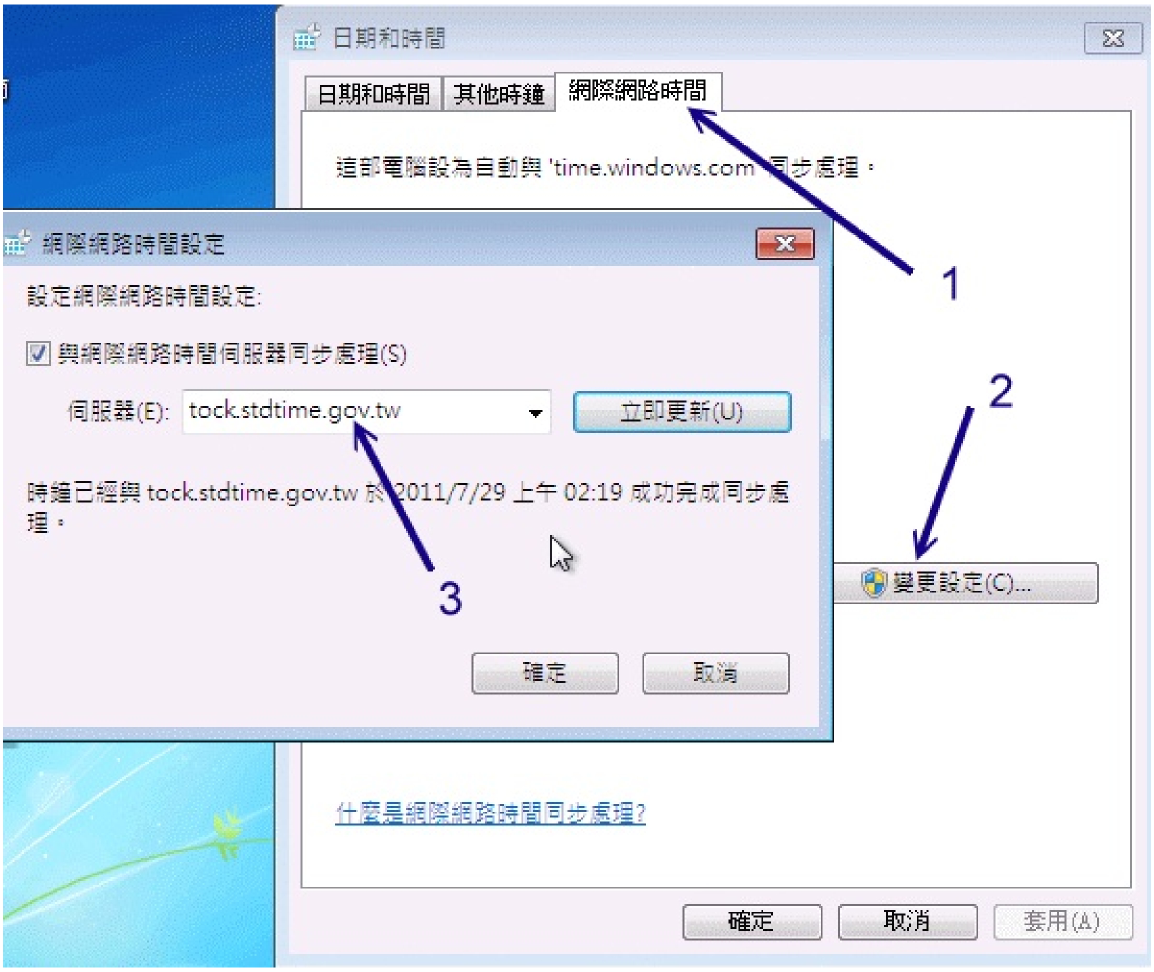


图 15.3-2、Windows 7 提供的网络校时功能

如上所示，你可以自行填写台湾的时间服务器来对应时间，当然也可以填写你自己的时间服务器啊！ 之后系统就会主动的上网去更新时间了。不过，这是 Windows XP 之后的窗口系统才有的功能，如果是比较早期的 Windows ， 例如 Windows 95/2000 预设是没有这个功能的。不过也没有关系， 因为国家频率与时间标准实验室 (http://www.stdtime.gov.tw/) 也有提供一个客户端软件喔！链接资料如下：

- http://www.stdtime.gov.tw/chinese/EXE/NTPClock.exe

你可以下载，直接执行他就知道如何处理了，因为是全中文接口的图形化软件嘛！

15.4 重点回顾

- 地球共有 24 个时区，而以格林威治时间 (GMT) 为标准时间；
- 台湾本地时间为 GMT + 8 小时；
- 最准确的时间为使用原子钟 (Atomic clock) 所计算的，例如 UTC (Coordinated Universal Time) 就是一例；
- Linux 系统本来就有两种时间，一种是 Linux 以 1970/01/01 开始计数的系统时间，一种则是 BIOS 记载的硬件时间；
- Linux 可以透过网络校时，最常见的网络校时为使用 NTP 服务器，这个服务启动在 udp port 123；
- 时区档案主要放置于 /usr/share/zoneinfo/ 目录下，而本地时区则参考 /etc/localtime；
- NTP 服务器为一种阶层式的服务，所以 NTP 服务器本来就会与上层时间服务器作时间的同步化， 因此 nptd 与 ntpdate 两个指令不可同时使用；
- NTP 服务器的联机状态可以使用 ntpstat 及 ntpq -p 来查询；
- NTP 提供的客户端软件为 ntpdate 这个指令；
- 在 Linux 下想要手动处理时间时，需以 date 设定时间后，以 hwclock -w 来写入 BIOS 所记录的时间。
- NTP 服务器之间的时间误差不可超过 1000 秒，否则 NTP 服务会自动关闭。

15.5 本章习题

- 什么是 GMT (格林威治) 时间与 UTC 时间？由于地球是圆的，所以同一时间点上，在地球共可分为 24 个时区，其中，我们以欧洲的格林威治时间为一个对照的依据，这个即是 GMT 时间。台湾时间比 GMT 时间快了 8 小时。至于 UTC 时间则是由原子钟所计算的时间，这个时间是相当的准确的， 主要仍以格林威治时间为时区！
- Linux 系统的所有时区档案放置哪一个目录底下？所有的时区档案放置于：/usr/share/zoneinfo 底下！至于系统时区的配置文件则在 /etc/sysconfig/clock 与 /etc/localtime 喔！
- 我的 Linux 主机本来放置在日本东京，现在想将他拿到台湾来运作，不过因为日本与台湾有一个小时的时差， 所以我的时间应该需要经过调整才行。不过，因为我的 BIOS Time 主要是依据 UTC 时间来设定的，所以似乎只要更动时区参数即可。请问我该如何设定时区，好让我的 Linux 主机能够显示正确的时间？先将 /etc/localtime 删除，然后将 /usr/share/zoneinfo/Asia/Taipei 这个档案复制成为 /etc/localtime 即可！
- 目前 Linux 系统上面的时间服务器主要是以 NTP 为主，请问这个 daemon 的主要配置文件放在哪里，而该配置文件中，针对上层 time server 的设定参数为何？而那个 driftfile 参数是干嘛用的？在 /etc/ntp.conf 这个档案当中，至于上层 time server 的设定参数为 server 啊！那个 driftfile 则是用来做为『时间差额』的计算的！该参数后面接的是一个完整路径的文件名，该档案里面的数值单位为百万分之一(ppm)。
- 请问 ntptrace 的功能为何？可以用来追踪上层 time server 的连接时间与目前时间！
- 我以 date 更新了我 Linux 上面的时间后，该如何将时间数据写入 BIOS 内？必须利用 hwclock 这个程序来写入，利用 hwclock -w 写入 BIOS
- 在 Linux 上面如何进行网络校时？最简单的方法即是使用『 ntpdate time.servers.ip && hwclock -w 』即可！

15.6 参考数据与延伸阅读

- 注1：格林威治时间的 Wiki 说明：http://en.wikipedia.org/wiki/Greenwich_Mean_Time
- 注2：UTC 时间的 Wiki 说明：http://en.wikipedia.org/wiki/Coordinated_Universal_Time
- 注3：台湾提供的几部标准时间服务器与时间服务器官网：
  http://www.stdtime.gov.tw/Time/ntp/resource.htm
  http://www.stdtime.gov.tw/Time/home.htm
- 注4：NTP 的官方网站：http://www.ntp.org 另一个好站：http://www.eecis.udel.edu/~mills/ntp/html/ntpd.html
- 由网友李涛兄提供的好站：
  http://support.ntp.org/bin/view/Support/TroubleshootingNTP#Section_9.5
  http://www.eecis.udel.edu/~mills/ntp/html/ntpq.html

2003/08/21：首次完成 2006/12/05：将旧的文章移动到此处 2006/12/08：将文章作了个版面修改，同时将一些数据再加强一些。并补充一些额外的查阅 NTP 的指令。 2009/04/28：将 offset 的时间单位写错了，应该是 ms 不是 us ，亦即 10^(-3) 秒，而不是 10^(-6) 秒。抱歉。 2011/02/18：由读者李涛兄提供的资料，发现原本 ntpq 的说明反了！ * 应为作用中，而 + 是较佳的另一部候选主机。 2011/03/16：将旧的基于 CentOS 4.x 的文章移动到 此处 2011/03/18：将一些重复性的数据汇整一下，不需要的部分就删除掉了～ 2011/07/28：将基于 CentOS 5.x 的文章移动到此处

第十六章、文件服务器之二： SAMBA 服务器

最近更新日期：2011/07/29

如果想要共享档案，在 Linux 对 Linux 的环境下，最简单的方法就是透过 NIS 这玩意儿了！至于 Windows 对 Windows 的环境下，最简单的方法则是『网络上的芳邻』啊。那如果你的区网中有 Windows 也有 Linux 而且想要共享文件系统的话，那该怎办？ 那就使用 Samba 服务器吧！Samba 可以让 Linux 加入 Windows 的网芳支持，让异质平台可以共享文件系统！非常好用的吻！不仅如此， Samba 也可以让 Linux 上面的打印机成为打印机服务器 (Printer Server)。鸟哥个人觉得， Samba 对于整个区网的贡献真的是很大啦！

16.1 什么是 SAMBA

在这个章节中，我们要教大家跳的是热情有劲的巴西 SAMBA 舞蹈... 喔不～搞错了～是要向大家介绍 SAMBA 这个好用的服务器啦！咦！怪了！ 怎么服务器的名称会使用 SAMBA 呢？还真是怪怪的呢！那么这个 SAMBA 服务器的功能是什么呢？另外， 它最早是经由什么样的想法而开发出来的呢？底下就让我们慢慢的谈一谈吧！

16.1.1 SAMBA 的发展历史与名称的由来

在早期的网络世界当中，档案数据在不同主机之间的传输大多是使用 FTP 这个好用的服务器软件来进行传送。不过使用 FTP 传输档案却有个小小的问题， 那就是你无法直接修改主机上面的档案数据！也就是说，你想要更改 Linux 主机上面的某个档案时，你必须要将该档案自服务器下载后才能修改。 也因此该档案在服务器与客户端都会存在。这个时候，万一如果有一天你修改了某个档案， 却忘记将数据上传回主机，那么等过了一阵子之后，呵呵，你如何知道那个档案才是最新的？

- 让档案在两部主机之间直接修改： NFS 与 CIFS

既然有这样的问题，那么好吧，我可不可以在客户端的机器上面直接使用服务器上面的档案， 如果可以在客户端直接进行服务器端档案的存取，那么我在客户端就不需要存在该档案数据啰，也就是说， 我只要有 Server 上面的档案资料存在就可以啦！有没有这样的文件系统啊！ 很高兴的是，第十三章的 NFS 就是这样的文件系统之一啦！我只要在客户端将 Server 所提供分享的目录挂载进来， 那么在客户端的机器上面就可以直接取用 Server 上的档案资料啰，而且， 该数据就像是我客户端上面的 partition 一般，真是好用！

而除了可以让 Unix Like 的机器互相分享档案的 NFS 服务器之外，在微软 (Microsoft) 操作系统上面也有类似的文件系统，那就是 Common Internet File System, CIFS 这个咚咚啦！CIFS 最简单的想法就是目前常见的『网络上的芳邻』咯！ Windows 系统的计算机可以透过桌面上『网络上的芳邻』来分享别人所提供的档案数据哩！真是方便。 不过，NFS 仅能让 Unix 机器沟通， CIFS 只能让 Windows 机器沟通。伤脑筋，那么有没有让 Windows 与 Unix-Like 这两个不同的平台相互分享档案数据的文件系统呢？

- 利用封包侦测逆向工程发展的 SMB Server

在 1991 年一个名叫 Andrew Tridgell 博士班研究生就有这样的困扰，他手上有三部机器，分别是跑 DOS 的个人计算机、DEC公司的 Digital Unix 系统以及 Sun 的 Unix 系统。在当时，DEC 公司有发展出一套称为 PATHWORKS 的软件，这套软件可以用来分享 DEC 的 Unix 与个人计算机的 DOS 这两个操作系统的档案数据，可惜让 Tridgell 觉得较困扰的是，Sun 的 Unix 无法藉由这个软件来达到数据分享的目的 (注1)。

这个时候 Tridgell 就想说：『咦！既然这两部系统可以相互沟通，没道理 Sun 就必需这么苦命吧？可不可以将这两部系统的运作原理找出来，然后让 Sun 这部机器也能够分享档案数据呢？』，为了解决这样的的问题，他老兄就自行写了个 program 去侦测当 DOS 与 DEC 的 Unix 系统在进行数据分享传送时所使用到的通讯协议信息，然后将这些重要的信息撷取下来，并且基于上述所找到的通讯协议而开发出 Server Message Block (SMB) 这个文件系统，而就是这套 SMB 软件就能够让 Unix 与 DOS 互相的分享数据啰！

Tips: 再次的给他强调一次，在 Unix Like 上面可以分享档案数据的 file system 是 NFS，那么在 Windows 上面使用的『网络上的芳邻』所使用的文件系统则称为 Common Internet File System, CIFS



- 取名 SAMBA 的主因 ^_^

既然写成了软件，想一想，总是需要注册一下商标吧！因此 Tridgell 就去申请了 SMBServer (Server Message Block 的简写) 这个名字来做为他撰写的这个软件的商标，可惜的是，因为 SMB 是没有意义的文字，因此没有办法达成注册。既然如此的话，那么能不能在字典里面找到相关的字词可以做为商标来注册呢？ 翻了老半天，呵呵！这个 **S**A**MB**A 刚好含有 SMB ，又是热情有劲的拉丁舞蹈的名称，不然就用这个名字来做为商标好了！这成为我们今天所使用的 SAMBA 的名称由来啦！ ^_^

16.1.2 SAMBA 常见的应用

由上面说明的 SAMBA 发展缘由，你就应该不难知道， SAMBA 最初发展的主要目就是要用来沟通 Windows 与 Unix Like 这两个不同的作业平台，那么 SAMBA 可以进行哪些动作呢？想一想网芳能做的吧！

- 分享档案与打印机服务；
- 可以提供用户登入 SAMBA 主机时的身份认证，以提供不同身份者的个别数据；
- 可以进行 Windows 网络上的主机名解析 (NetBIOS name)
- 可以进行装置的分享 (例如 Zip, CDROM...)

底下我们来谈几个 SAMBA 服务器的应用实例吧！

- 利用软件直接编修 WWW 主机上面的网页数据

相信很多人都是利用个人计算机将网页制作完毕之后，再以类似 FTP 之类的服务将网页上传到 WWW 主机的， 但这样有个困扰，那就是同时在客户端与 WWW 主机上头都有一份网页数据，常常会忘记哪一份是最新的， 最麻烦的是，有时候下载下来的档案已经经过好多修改

了，却在下次的 FTP 作业，不小心又下载一次旧数据， 结果将已经修改过的数据覆盖过去～ 天呐！又要重写一遍.....真是讨厌！

如果你有安装 SAMBA 服务器的设定的话，那么透过『网芳』的功能，直接联机远程服务器所提供的目录， 如此一来你可以直接在你的个人计算机上面修改主机的档案数据，只有一份正确的数据而已喔！ 这就有点像是『在线编修』呢，一修改完成，在 Internet 上面可以立刻检验，方便的很呐！

- 做成可直接联机的文件服务器

在鸟哥过去待过的实验室中，由于计算机数量不多，研究生常常会使用到不同的计算机 (因为大家都得抢没有人用的计算机啊！) ，此外，也常常有研究生拿自己的 NoteBook 来工作，因此，有些团队的数据就分散在各个计算机当中，使用上相当的不方便。 这个时候，鸟哥就使用 SAMBA 将硬盘空间分享出来，由于使用者要登入 SAMBA 这个服务器主机时需要输入用户数据 (账号与密码)，而不同的登入者会取得不一样的目录资源，所以可以避免自己的数据在公用计算机上面被窥视， 此外，在不同的公用计算机上面都可以登入 SAMBA 主机，数据的使用上面真是相当的棒啊！

- 打印机服务器

SAMBA 除了分享文件系统外，也可以分享打印机喔，鸟哥的研究室好几部计算机就是直接以 Linux 分享的打印机来印制报告的。你会说『啊 Windows 也可以办的到啊！没有什么了不起的！』是啊。 但是鸟哥认为，用 Linux 做为服务器主机时毕竟还是比较稳定一点，可以 24 小时且全年无休的努力工作呐。此外，因为目前透过『网络上的芳邻』来攻击局域网络的 Windows 操作系统的计算机病毒实在是太多了，防不胜防， Linux 对于这样的攻击并没有很大的影响 (因为常见的攻击手法均针对 Windows 而来～)，所以也比较安全一些说～

SAMBA 的应用挺广泛的，尤其对于局域网络内的计算机来说，更是一项不可多得的好用的服务器， 虽然或许你会说，SAMBA 的功能不过是模仿 Windows 的网芳以及 AD 相关的软件，那我直接使用 Windows 不就 OK 了？可惜的是， Windows XP 对于网芳的联机限制依版本而有所不同， 以企业常见的专业版 (Professional) 来说，他仅能提供最多同时十个联机到网芳的联机能力，这...不太够用吧！ 所以啰，SAMBA 稳定、可靠又没有限制联机数，值得学习吧！^_^！更多的应用你可以自行发掘呐！

16.1.3 SAMBA 使用的 NetBIOS 通讯协议

事实上，就像 NFS 是架构在 RPC Server 上面一样， SAMBA 这个文件系统是架构在 NetBIOS (Network Basic Input/Output System, NetBIOS) 这个通讯协议上面所开发出来的。 既然如此，我们当然就要了解一下 NetBIOS 啰！

最早 IBM 发展出 NetBIOS 的目的仅是要让局域网络内少数计算机进行网络链接的一个通讯协议而已， 所以考虑的角度并不是针对大型网络，因此，这个 NetBIOS 是无法跨路由的 (Router / Gateway)。这个 NetBIOS 在局域网络内实在是很好用， 所以微软的网络架构就使用了这个咚咚来进行沟通的呐！而 SAMBA 最早发展的时候，其实是想要让 Linux 系统可以加入 Windows 的系统当中来分享使用彼此的档案数据的，所以当然 SAMBA 就架构在 NetBIOS 发展出来啰。

不过 NetBIOS 是无法跨路由的，因此使用 NetBIOS 发展起来的服务器理论上也是无法跨越路由的呢！ 那么该服务器的使用范围不就受限相当的多了？好在，我们还有所谓的 NetBIOS over TCP/IP 的技术呢！这是什么样的技术啊？

举个例子来说好了，我们知道 TCP/IP 是目前网络连接的基本协议，现在我们将 NetBIOS 想成是一封明信片， 这个明信片只能让你自己欣赏而已，如果今天我们要将这个明信片送到远方的朋友那边时！就需要透过邮件系统 (例如邮局啦、国际快递啦等等的) 来传送了！这个 TCP/IP 就可以视为邮件传递系统啦！透过这个 NetBIOS over TCP/IP 的技术，我们就可以跨路由的使用 SAMBA 服务器所提供的功能咯！ 当然啦，目前 SAMBA 还是比较广泛的使用在 LAN 里面说。

Tips: 或许你会发现在 Windows 网络设定里面常常看到 NetBEUI 这个咚咚，那是什么呢？那个是 NetBIOS Extened User Interface 的简写，也是 IBM 在 NetBIOS 发展出来之后的改良版本。虽然这两者的技术不太相同， 不过，我们只要知道一些简单的概念就可以了！所以，在这里我们不针对 NetBEUI 来介绍。



16.1.4 SAMBA 使用的 daemons

NetBIOS 当初发展时就着眼在局域网络内的快速数据交流，而因为是定义在局域网络内，因此他并没有使用类似 TCP/IP 之类的传输协议，也就不需要 IP 的设定。如此一来数据如何在两部主机之间交流呢？ 其实主机在 NetBIOS 协议当中的定义为使用『NetBIOS Name』，每一部主机必须要有不同的 NetBIOS Name 才行， 而档案数据就是在不同的 NetBIOS name 之间沟通啰！我们以一个网芳的设定来作简单的说明好了：

1. 取得对方主机的 NetBIOS name 定位该主机所在：

   当我们想要登入某部 Windows 主机使用他所提供的档案数据时，必需要加入该 Windows 主机的群组 (Workgroup)，并且我们的机器也必需要设定一个主机名，注意喔，这个主机名跟 Hostname 是不一样的，因为这个主机名是架构在 NetBIOS 协议上

的，我们可以简单的称呼他为 NetBIOS Name。在同一个群组当中，NetBIOS Name 必需要是独一无二的喔！

2. 利用对方给予权限存取可用资源：

   在我们找到该主机名后，是否能登入该对方主机或者是取用对方主机所提供的资源， 还要看对方 Windows 主机有没有提供我们使用的权限呐！所以，并不是登入该 Windows 主机之后我们就可以无限制的取用该主机的档案资源了。也就是说，如果对方主机允许你登入， 但是却没有开放任何资源让你取用，呵呵，登入主机也无法查看对方的硬盘里面的数据的啦！

我们的 SAMBA 则是透过两支服务来控制这两个步骤，分别是：

- nmbd ：这个 daemon 是用来管理工作组啦、NetBIOS name 啦等等的解析。主要利用 UDP 协议开启 port 137, 138 来负责名称解析的任务；
- smbd ：这个 daemon 的主要功能就是用来管理 SAMBA 主机分享的目录、档案与打印机等等。 主要利用可靠的 TCP 协议来传输数据，开放的端口为 139 及 445(不一定存在) 。

所以啰， SAMBA 每次启动至少都需要有这两个 daemons 喔！这可不要忘记啰！而当我们启动了 SAMBA 之后，主机系统就会启动 137, 138 这两个 UDP 及 139 这一个 TCP 埠口，这也不要忘记了！ 因为后面设定防火墙的时候，还会使用到这三个 port 的呢！

16.1.5 联机模式的介绍 (peer/peer, domain model)

SAMBA 服务器的应用相当的广泛，而且可以依照不同的网域联机方式，与不同的用户账号密码的控管方式来进行分类。 例如最常见的 Workgroup 及 Domain 两种方式的联机模式呢！底下我们就是要来谈一谈这两种最常见的局域网络的联机模式： peer/peer (对等模式) 及 domain model (主控模式)。

- peer/peer (Workgroup model, 对等模式)：

peer 有同等、同辈的意思存在，所以由字面上来看，peer/peer 当然就是指两部主机的地位相等啰！ 这是什么意思呢？简单的说，假如在局域网络里面的所有 PC 均可以在自己的计算机上面管理自己的账号与密码， 同时每一部计算机也都具有独力执行各项软件的能力，只是藉由网络将各个 PC 链接在一起而已的一个架构， 所以，每一部机器都是可以独立运作的喔！

这样的架构在目前小型办公室里面是最常见的。例如办公室里面有十个人，每个人桌上可能都安装有一套 Windows 操作系统的个人计算机，而这十部计算机都可以独立进行办公室软件的执行啊、独立上网啊、独立玩游戏啊等等的， 因为这十部计算机都可以独立运作，所以不会有一部计算机关掉，其他的计算机就无法工作的情况发生，这就是 peer/peer 的典型架构。

那在这样的架构底下，要如何透过网络联机来取得对方的数据呢？举例来说，以下图的架构为例，在这样的架构下，假设 vbird (PC A) 写了一个报告书，而 dmtsai (PC B) 想要以网络直接取用这个报告书时，那 dmtsai 就必须要知道 vbird 使用的密码，并且 vbird 必须要在 PC A 上面启用 Windows 的『资源共享(或者是共享)』之后，才能够让 dmtsai 联机进入喔 (此时 PC A 为 Server) ！而且， vbird 可以随时依照自己的喜好来更改自己的账号与密码，而不受 dmtsai 的影响。不过，dmtsai 就得要取得 vbird 同意取得新的账号与密码后，才能够登入 PC A 喔！反过来说，同样的， vbird 要取得 dmtsai 的数据时，同样需要取得 PC B 的账号与密码后，才能够顺利登入啊 (此时 PC A 为 Client 喔)！因为 PC A, PC B 的角色与地位都同时可以为 Client 与 Server ，所以就是 peer/peer 的架构了！



图 16.1-1、peer/peer 联机的示意图

使用 peer/peer 的架构的好处是每部计算机均可以独立运作，而不受他人的影响！不过， 缺点就是当整个网域内的所有人员都要进行数据分享时，光是知道所有计算机里面的账号与密码，就会很伤脑筋了！ 所以， Peer/Peer 的架构是比较适合 (1)小型的网域，或者是 (2)没有需要常常进行档案数据分享的网络环境，或者是 (3)每个使用者都独自拥有该计算机的拥有权 (就是说，该计算机是用户的，而不是公用的啦！) 而，如果该单位的所有 PC 均是公有的 (例如学校的计算机教室环境)，而且你需要统一控管整个网域里面的账号与密码的话， 那就得使用底下的 domain models 了！

- domain model (主控模式)

假设今天你服务的单位有 10 部计算机，但是你的单位有 20 个员工，这也就是说，这 20 个员工轮流抢着用这 10 部计算机。如果每部计算机都如同 peer/peer 的架构时，那么每部计算机都需要输入这 20 个员工的账号与密码来提供他们登入喔。而且，今天假如有个员工想要变更自己的密码时，就需要到 10 台计算机上面进行密码变更的作业！否则他就必须要记得这 10 部计算机里面，那一部计算机是记忆那一个密码...好烦那～

如果上述是这样的情况，使用 peer/peer 架构就不是一个好方法了！这个时候就需要藉由 domain model 来达成你的需求啦！所谓的 domain model 概念其实也很简单，既然使用计算机资源需要账号与密码， 那么我将所有的账号与密码都放置在一部主控计算机 (Primary Domain Controller, PDC) 上面，在我的网域里面，任何人想要使用任何计算机时，都需要在屏幕前方输入账号与密码，然后通通藉由 PDC 服务器的辨识后，才给予适当的权限。也就是说，不同的身份还具有不一样的计算机资源权限就是了！ 例如底下的图示：

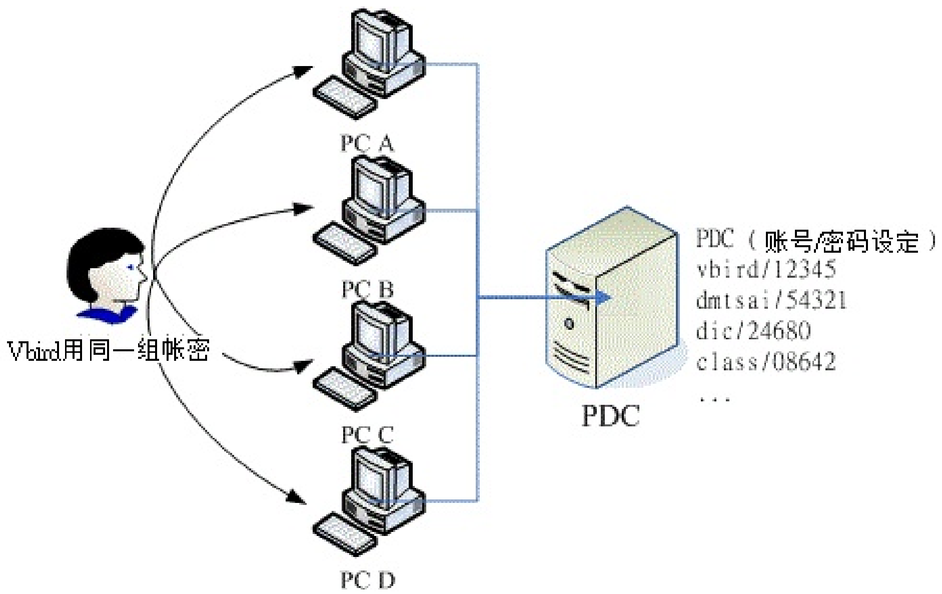


图 16.1-2、domain model 联机的示意图

PDC 服务器控管整个网域里面的各个机器 (PC A ~ PC D) 的账号与密码的信息，假如今天有个使用者账号名称为 vbird ，且密码为 12345 时，他不论使用哪一部计算机 (PC A ~ PC D) 只要在屏幕前方输入 vbird 与他的密码，则该机器会先到 PDC 上面查验是否有 vbird 以及 vbird 的密码，并且 PDC 主机会给予 vbird 这个用户相关的计算机资源权限。当 vbird 在任何一部主机上面登入成功后，他就可以使用相关的计算机资源了！

这样的架构比较适合人来人往的企业架构，当系统管理员要控管新进人员的计算机资源使用权时，可以直接针对 PDC 来修改就好了，不需要每一部主机都去修修改改的，对于系统管理员来说，这样的架构在控管账号资源上，当然是比较简单的啦！

各种架构适用的环境与适用的人都不相同，并没有那个是最好啦！请依照你的工作环境来选择联机的模式啰！当然， SAMBA 可以达到上述两种模式的啦！底下我们会分别来介绍喔！

16.2 SAMBA 服务器的基础设定

SAMBA 这个软件几乎在所有的 Linux distributions 上面都有提供，因为即使你的 Linux 仅做为个人桌面计算机使用时， 你依旧可能会需要联机到远程的 Windows 网芳，那个时候就得要 samba 提供的客户端软件功能啰！ 因此你只要直接安装系统上面提供的默认 samba 版本即可。底下我们会先介绍 samba 服务器，然后再介绍客户端功能喔！

16.2.1 Samba 所需软件及其软件结构

目前常见的 samba 版本为 3.x 版，旧版的 2.x 版在设定上有点不一样，因此在进入设定前请先确认你的 samba 版本。 咱们的 CentOS 6.x 主要提供的是 Samba 3.x 的版本，不过也有释出 4.x 的版本 (samba4)，我们这里主要介绍的是预设的 3.x 版本的。那么你需要什么软件呢？基本上有这些：

- samba： 这个软件主要提供了 SMB 服务器所需的各项服务程序 (smbd 及 nmbd)、 的文件档、以及其他与 SAMBA 相关的 logrotate 配置文件及开机默认选项档案等；
- samba-client： 这个软件则提供了当 Linux 做为 SAMBA Client 端时，所需要的工具指令，例如挂载 SAMBA 文件格式的 mount.cifs、 取得类似网芳相关树形图的 smbtree 等等；
- samba-common： 这个软件提供的则是服务器与客户端都会使用到的数据，包括 SAMBA 的主要配置文件 (smb.conf)、语法检验指令 (testparm) 等等；

这三个软件你都得要安装才行喔！如果尚未安装的话，使用 yum 去装好它吧！安装完毕之后，你可以依序察看一下 Samba 的软件结构喔！与它相关的配置文件基本上有这些：

- /etc/samba/smb.conf： 这是 Samba 的主要配置文件，基本上，咱们的 Samba 就仅有这个配置文件而已，且这个配置文件本身就是很详细的说明文件了，请用 vim 去查阅它吧！主要的设定项目分为服务器的相关设定 (global)，如工作组、NetBIOS 名称与密码等级等， 以及分享的目录等相关设定，如实际目录、分享资源名称与权限等等两大部分。
- /etc/samba/lmhosts： 早期的 NetBIOS name 需额外设定，因此需要这个 lmhosts 的 NetBIOS name 对应的 IP 檔。 事实上它有点像是 /etc/hosts 的功能！只不过这个 lmhosts 对应的主机名是 NetBIOS name 喔！不要跟 /etc/hosts 搞混了！目前 Samba 预设会去使用你的本机名称 (hostname) 作为你的 NetBIOS name，因此这个档案不设定也无所谓。
- /etc/sysconfig/samba： 提供启动 smbd, nmbd 时，你还想要加入的相关服务参数。

- /etc/samba/smbusers： 由于 Windows 与 Linux 在管理员与访客的账号名称不一致，例如： administrator (windows) 及 root(linux)， 为了对应这两者之间的账号关系，可使用这个档案来设定
- /var/lib/samba/private/{passdb.tdb,secrets.tdb}： 管理 Samba 的用户账号/密码时，会用到的数据库档案；
- /usr/share/doc/samba-<版本>： 这个目录包含了 SAMBA 的所有相关的技术手册喔！也就是说，当你安装好了 SAMBA 之后，你的系统里面就已经含有相当丰富而完整的 SAMBA 使用手册了！值得高兴吧！ ^_^，所以，赶紧自行参考喔！

至于常用的脚本文件案方面，若分为服务器与客户端功能，则主要有底下这几个数据：

- /usr/sbin/{smbd,nmbd}：服务器功能，就是最重要的权限管理 (smbd) 以及 NetBIOS name 查询 (nmbd) 两个重要的服务程序；
- /usr/bin/{tdbdump,tdbtool}：服务器功能，在 Samba 3.0 以后的版本中，用户的账号与密码参数已经转为使用数据库了！Samba 使用的数据库名称为 TDB (Trivial DataBase)。既然是使用数据库，当然要使用数据库的控制指令来处理啰。tdbdump 可以察看数据库的内容，tdbtool 则可以进入数据库操作接口直接手动修改帐密参数。不过，你得要安装 tdb-tools 这个软件才行；
- /usr/bin/smbstatus：服务器功能，可以列出目前 Samba 的联机状况， 包括每一条 Samba 联机的 PID, 分享的资源，使用的用户来源等等，让你轻松管理 Samba 啦；
- /usr/bin/{smbpasswd,pdbedit}：服务器功能，在管理 Samba 的用户账号密码时， 早期是使用 smbpasswd 这个指令，不过因为后来使用 TDB 数据库了，因此建议使用新的 pdbedit 指令来管理用户数据；
- /usr/bin/testparm：服务器功能，这个指令主要在检验配置文件 smb.conf 的语法正确与否，当你编辑过 smb.conf 时，请务必使用这个指令来检查一次，避免因为打字错误引起的困扰啊！
- /sbin/mount.cifs：客户端功能，在 Windows 上面我们可以设定『网络驱动器机』来连接到自己的主机上面。在 Linux 上面，我们则是透过 mount (mount.cifs) 来将远程主机分享的档案与目录挂载到自己的 Linux 主机上面哪！
- /usr/bin/smbclient：客户端功能，当你的 Linux 主机想要藉由『网络上的芳邻』的功能来查看别台计算机所分享出来的目录与装置时，就可以使用 smbclient 来查看啦！这个指令也可以使用在自己的 SAMBA 主机上面，用来查看是否设定成功哩！
- /usr/bin/nmblookup：客户端功能，有点类似 nslookup 啦！重点在查出 NetBIOS name 就是了。

- /usr/bin/smbtree：客户端功能，这玩意就有点像 Windows 系统的网络上的芳邻显示的结果，可以显示类似『靠近我的计算机』之类的数据， 能够查到工作组与计算机名称的树状目录分布图！

大致的软件结构就是这样，底下就准备来讲一个简单的案例吧！这样比较好介绍配置文件项目啦！

16.2.2 基础的网芳分享流程与 smb.conf 的常用设定项目

既然 Samba 是要加入 Windows 的网芳服务当中，所以它的设定方式应该是要与网芳差不多才是。所以我们先来聊一聊 Windows 的一些网芳设定方法再说。在早期 Windows 的网芳设定眞是很简单，不过也因为太简单， 所以产生的安全问题可是相当的麻烦的。后来在 Windows XP 的 SP2 (服务包第二版) 之后加入了很多的预设防火墙机制， 因此使用网芳的预设限制常常会是这样的：

- 服务器与客户端之间必须要在同一个网域当中 (否则需要修改 Windows 预设防火墙)；
- 最好设定为同一工作组；
- 主机的名称不可相同 (NetBIOS name)；
- 专业版 Windows XP 最多仅能提供同时 10 个用户联机到同一台网芳服务器上。

工作组与主机名的设定，你可以在『我的计算机』右键单击，选择内容后去修订相关的设定值。当你的 Windows 主机群符合上述的条件后，就很容易处理网芳分享的工作啦！分享的步骤一般是这样的：

1. 叫出档案总管，然后在要分享的目录、磁盘或装置 (如打印机) 上面按下右键，选择『共享』，然后就能够设定好分享的数据了；
2. 最好建立一组给用户使用的账号与密码，让其他主机的用户可以透过该账号密码联机进入使用网芳分享的资源；

例题：假设你打开 Windows XP 的档案总管，在 D:\VBird\Data 这个目录下，你按下右键选『共享与安全性』，之后，在出现的窗口中， 你选择：『你了解这个安全风险，但仍不要执行精灵而共享档案，请按这里』，然后勾选：『在网络上共享这个文件夹』， 最后共享的名称你输入了：『VBGame』，请问，假设你的 IP 是 192.168.100.20 ，那么你的用户会看到什么网址列？答：网芳的资源名称通常的写法是：『 \IP\分享资源名称』，我们的分享资源名称为 VBGame，因此最终这个分享的资源名称应该是：『 \192.168.100.20\VBGame 』才对！很多朋友都会写成：『 \192.168.100.20\VBird\Game 』那错得很离谱喔！

眞是有够简单的！那么 Samba 怎么设定啊？也是很简单，依据上述的限制以及流程你可以这样想象：

1. 服务器整体设定方面：在 smb.conf 当中设定好工作组、NetBIOS 主机名、密码使用状态 (无密码分享或本机密码) 等等；

2. 规划准备分享的目录参数：在 smb.conf 内设定好预计要分享的目录或装置以及可供使用的账号数据；
3. 建立所需要的文件系统：根据步骤 2 的设定，在 Linux 文件系统当中建立好分享出去的档案或装置，以及相关的权限参数；
4. 建立可用 Samba 的账号：根据步骤 2 的设定，建立所需的 Linux 实体账号，再以 pdbedit 建立使用 Samba 的密码；
5. 启动服务：启动 Samba 的 smbd, nmbd 服务，开始运转哩！

根据上面的流程，其实我们最需要知道的就是 smb.conf 这个配置文件的信息就是了。 所以首先我们就要来介绍一下这个档案的设定方式啰！这个档案其实可以分为两部份来看， 一个是主机信息部分，在 smb.conf 当中以 [global] (全领域) 作为设定的依据；另一个则是分享的信息， 以个别的目录名称为依据。另外，由于 Samba 主要是想加入网芳功能，因此在 smb.conf 内的很多设定都与 Windows 类似喔：

- 在 smb.conf 当中，井字号与分号 (# 跟 ;) 都是批注符号；
- 在这个配置文件中，大小写是没关系的！因为 Windows 没分大小写！

- smb.conf 的服务器整体参数： [global] 项目

在 smb.conf 这个配置文件当中的设定项目有点像底下这样：

```
# 会有很多加上 # 或 ; 的批注说明，你也可以自行加上来提醒自己相关设定
[global]
    参数项目 = 设定内容
    ....

[分享资源名称]
    参数项目 = 设定内容
    ....
```

在 [global] 当中的就是一些服务器的整体参数了，包括工作组、主机的 NetBIOS 名称、字符编码的显示、登录文件的设定、 是否使用密码以及使用密码验证的机制等等，都是在这个 [global] 项目中设定的。至于 [分享资源名称] 则是针对你开放的目录来进权限方面的设定，包括谁可以浏览该目录、是否可以读写等等参数。 在 [global] 部分关于主机名信息方面的参数主要有：

- workgroup = 工作组的名称：注意，主机群要相同；
- netbios name = 主机的 NetBIOS 名称啊，每部主机均不同；
- server string = 主机的简易说明，这个随便写即可。

另外，过去常常让使用者心生不满的语系显示问题方面，你务必要清楚的知道的是，SAMBA 服务器上面的数据 (例如 mount 磁盘分区槽的参数以及原本的数据编码), SAMBA 服务器显示的语系, Windows 客户端显示的语系, Windows 客户端连上 SAMBA 的软件 都需要符合设定值才行！在新版的 3.x 上面有数个提供这些语系转换的设定喔，如下所示：

- display charset = 自己服务器上面的显示编码， 例如你在终端机时所查阅的编码信息。一般来说，与底下的 unix charset 会相同。

- unix charset = 在 Linux 服务器上面所使用的编码，一般来说就是 i18n 的编码啰！ 所以你必须要参考 /etc/sysconfig/i18n 内的『默认』编码。
- dos charset = 就是 Windows 客户端的编码了！ 一般来说我们的繁体中文 Windows 使用的是 big5 编码，这个编码在 Samba 内的格式被称为『 cp950 』喔！

关于语系编码，建议你参考一下讨论区的这一篇：

- http://phorum.vbird.org/viewtopic.php?t=22001

我们的网友 eyesblue 写得太好了！所以建议大家直接前往查阅即可！在这里鸟哥将该文章内容作个例题来玩玩。

例题：假设你的 Samba 使用的语系 /etc/sysconfig/i18n 显示的是『 LANG="zh_TW.big5" 』， 而预计要分享的目标 Windows 系统是 XP，那么你的语系数据应该如何设定？答：由于 Linux, Windows XP 都使用 big5 编码，因此设定值应该是：

```
unix charset    = cp950
display charset = cp950
dos charset     = cp950
```

除此之外，还有登录文件方面的信息，包括这些参数：

- log file = 登录档放置的档案，文件名可能会使用变量处理；
- max log size = 登录档最大仅能到多少 Kbytes ，若大于该数字，则会被 rotate 掉。

还有网芳开放分享时，安全性程度有关的密码参数，包括这几个：

- security = share, user, domain：三选一，这三个设定值分别代表：
    - share：分享的数据不需要密码，大家均可使用 (没有安全性)；
    - user ：使用 SAMBA 服务器本身的密码数据库，密码数据库与底下的 passdb backend 有关；
    - domain：使用外部服务器的密码，亦即 SAMBA 是客户端之意，如果设定这个项目， 你还得要提供『password server = IP』的设定值才行；
- encrypt passwords = Yes 代表密码要加密，注意那个 passwords 要有 s 才对！
- passdb backend = 数据库格式，如前所述，为了加快速度， 目前密码文件已经转为使用数据库了！默认的数据库格式微 tdbsam ，而预设的档案则放置到 /var/lib/samba/private/passwd.tdb。

事实上 Samba 的密码方面设定值很多喔，包括你还可以利用 samba 来修改 /etc/passwd 里头的人物的密码呢！ 不过这个时候就得需要『 unix password sync 』以及『 passwd program 』这两个参数值的帮忙了。 我们这里先谈比较简单的，其他进阶的部分可以 man smb.conf 去进行搜寻查阅喔！ ^_^

- 分享资源的相关参数设定 [分享的名称]

这部分就是我们在前面的小范例当中说明的，要将 (1)哪个实际的目录 (2)分享成什么名称？中刮号里面放的是『分享名称』！ 那在这个分享名称内常见的参数有：

- [分享名称] ：这个分享名称很重要，它是一个『代号』而已。记得回去看看 16.2.2 里面提到的那个范例；
- comment ：只是这个目录的说明而已！
- path ：这个分享名称实际会进入的 Linux 文件系统 (目录)。 也就是说，在网芳当中看到的是 [分享] 的名称，而实际操作的文件系统则是在 path 里头所设定的。
- browseable ：是否让所有的用户看到这个项目？
- writable ：是否可以写入？这里需要注意一下喔！那个 read only 与 writable 不是两个蛮相似的设定值吗？如果 writable 在这里设定为 yes ，亦即可以写入，如果 read only 同时设定为 yes ， 那不就互相抵触了！那个才是正确的设定？答案是：最后出现的那个设定值为主要的设定！
- create mode 与 directory mode 都与权限有关的咯！
- writelist = 使用者, @群组，这个项目可以指定能够进入到此资源的特定使用者。 如果是 @group 的格式，则加入该群组的使用者均可取得使用的权限，设定上会比较简单！

因为分享的资源主要与 Linux 系统的档案权限有关，因此里头的设定参数多与权限有关。

- smb.conf 内的可用变量功能

为了简化设定值，Samba 提供很多不同的变量给我们来使用，主要有底下这几个变量喔：

- %S：取代目前的设定项目值，所谓的『设定项目值』就是在 [分享] 里面的内容！举例来说，例如底下的设定范例：

  ```
  [homes]
     valid users = %S
     ....
  ```

  因为 valid users 是允许的登入者，设定为 %S 表示任何可登入的使用者都能够登入的意思～今天如果 dmtsai 这个使用者登入之后，那个 [homes] 就会自动的变成了 [dmtsai] 了！这样可以明白了吗？ %S 的用意就是在替换掉目前 [ ] 里面的内容啦！

- %m：代表 Client 端的 NetBIOS 主机名喔！
- %M：代表 Client 端的 Internet 主机名喔！就是 HOSTNAME。
- %L：代表 SAMBA 主机的 NetBIOS 主机名。
- %H：代表用户的家目录。
- %U：代表目前登入的使用者的使用者名称
- %g：代表登入的使用者的组名。
- %h：代表目前这部 SAMBA 主机的 HOSTNAME 喔！注意是 hostname 不是 NetBIOS

name 喔！
- %I：代表 Client 的 IP 咯。
- %T：代表目前的日期与时间

以上就是在 smb.conf 上头常看到的几种设定项目，相信初次接触 Samba 的朋友，看到上头写的资料肯定是一头雾水的！ 我们底下用几个小范例来实际的介绍 smb.conf 的设定后，你就会知道这些参数如何应用了！ 记得，看完底下的下范例后，要回来再将这些参数的意义瞧一瞧， 而且若有其他额外的参数须知，务必自行 man smb.conf 喔！重要的很！

Tips: 时代变动太快，版本变动太多～要讲完所有的参数实在是很难的一件事～所以在这里鸟哥只讲一些常用的设定项目， 很多细项就得要靠各位看官自己努力了～文末也有列出很多 Samba 的在线资源，记得要查察看！



16.2.3 不需密码的分享 (security = share, 纯测试)

瞎密？不需要密码就能够使用 SAMBA 主机所提供的目录资源？真假？没错啦，可以达到的。不过，因为不需要密码就能够登入， 虽然你可以设定权限成为只读，让使用者可以『瞧瞧而已』，但是毕竟比较危险。 因为如果你不小心将重要数据放置到该分享的目录当中，岂不危险？所以尽量不要这样设定，所以标题才会讲：『纯测试』嘛！

- 0. 假设条件

在底下的案例中，服务器 (192.168.100.254) 预计设定的参数状况为：

- 在 LAN 内所有的网芳主机工作组 (workgroup) 为： vbirdhouse
- 这部 Samba 服务器的 NetBIOS 名称 (netbios name) 为： vbirdserver
- 使用者认证层级设定 (security) 为： share
- 取消原本有放行的 [homes] 目录；
- 仅分享 /tmp 这个目录而已，且取名为： temp
- Linux 服务器的编码格式假设为万国码 (Unicode, 亦即 utf8)
- 客户端为中文 Windows ，在客户端的软件也使用 big5 的编码

老实说， netbios name 几乎可以不用设定了，因为现在我们都用 IP 进行网芳联机，不一定会使用主机名嘛！ 所以这一版当中，鸟哥取消了 lmhosts 的设定值喔！好了，底下就开始依序来进行 samba 的设定吧！

- 1. 设定 smb.conf 配置文件

由于我们有设定语系相关的数据，因此得要先查查看，到底我们 Linux 服务器的语系是否为 utf8 呢？检查方法如下：

```
[root@www ~]# cat /etc/sysconfig/i18n
LANG="zh_TW.UTF-8"  &lt;==确实是出现了 utf8 喔！
```

如上所示，确实是 utf8 啊！而在这个例子当中我们仅分享 /tmp 这个目录而已，而且假设这个分享出来的目录是可擦写的，另外，我们并没有分享打印机喔！而在 smb.conf 当中的批注符号可以是『#』也可以是『;』喔！要注意！

```
[root@www ~]# cd /etc/samba
[root@www samba]# cp smb.conf smb.conf.raw  &lt;==先备份再说！
[root@www samba]# vim smb.conf
# 1\. 先设定好服务器整体环境方面的参数
[global]
        # 与主机名有关的设定信息
        workgroup     = vbirdhouse
        netbios name  = vbirdserver
        server string = This is vbird's samba server

        # 与语系方面有关的设定项目喔，为何如此设定请参考前面的说明
        unix charset    = utf8
        display charset = utf8
        dos charset     = cp950

        # 与登录文件有关的设定项目，注意变量 (%m)
        log file = /var/log/samba/log.%m
        max log size = 50

        # 这里才是与密码有关的设定项目哩！
        **security = share**

        # 修改一下打印机的加载方式，不要加载啦！
        load printers    = no

# 2\. 分享的资源设定方面：主要得将旧的批注，新的加入！
#    先取消 [homes], [printers] 的项目，然后针对 /tmp 的设定，可浏览且可写入喔
[temp]                                     &lt;==分享资源名称
        comment    = Temporary file space  &lt;==简单的解释此资源
        path       = /tmp                  &lt;==实际 Linux 分享的目录
        writable   = yes                   &lt;==是否可写入？在此例为是的
        browseable = yes                   &lt;==能不能被浏览到资源名称
        &lt;u&gt;guest ok   = yes&lt;/u&gt;;                   &lt;==单纯分享时，让用户随意登入
```

请你特别留意，在原本的 smb.conf 上面就已经有很多默认值了，这些默认值如果你不知道他的用途，尽量保留默认值，也可以使用 man smb.conf 去查询该默认值的意义。上述的设定是完全控制使用者的认证层级的呦！

- 2. 用 testparm 查阅 smb.conf 的语法设定正确性

在启动 samba 之前，我们务必要了解到 smb.conf 里面语法是否正确，检验的方式使用 testparm 这个指令即可。 测试方式如下：

```
[root@www ~]# testparm
选项与参数：
-v ：查阅完整的参数设定，连同默认值也会显示出来喔！

[root@www ~]# testparm
Load smb config files from /etc/samba/smb.conf
Processing section "[temp]"   &lt;==看有几个中括号，若中刮号前出现讯息，则有错误
Loaded services file OK.
Server role: ROLE_STANDALONE
Press enter to see a dump of your service definitions &lt;==按 Enter 继续

[global]   &lt;==底下就是刚刚在 smb.conf 里头设定的数据！
        dos charset = cp950
        unix charset = utf8
        display charset = utf8
        workgroup = VBIRDHOUSE
        netbios name = VBIRDSERVER
        server string = This is vbird's samba server
        security = SHARE
        log file = /var/log/samba/log.%m
        max log size = 50
        load printers = No

[temp]
        comment = Temporary file space
        path = /tmp
        read only = No
        guest ok = Yes
```

上头是语法验证与各个项目的列出，如果你下达 testparm 却出现如下画面那就是有问题：

```
[root@www ~]# testparm
Load smb config files from /etc/samba/smb.conf
Unknown parameter encountered: "linux charset" &lt;==中括号前为错误讯息！
Ignoring unknown parameter "linux charset"
Processing section "[temp]"
Loaded services file OK.
Server role: ROLE_STANDALONE
Press enter to see a dump of your service definitions
```

如果发现上述的错误，这表示你的 smb.conf 有个『 linux charset 』的设定参数，不过 smb.conf 其实是不支持这个参数的。 可能的问题是 samba 2.x 与 samba 3.x 有一些项目的支持已经不存在了，所以你使用旧版的 2.x 配置文件来 3.x 上头执行时，就会出现问题。此外，『打字错误』也是很常见的一个问题呐！赶紧测试一下语法先， 然后根据 smb.conf 存在的项目去进行修改吧。

如果你想要了解 samba 的所有设定 (包括没有在 smb.conf 里头设定的默认值)，可以使用 testparm -v 来作详细的输出， 资料相当的丰富，透过这个你也可以知道你的主机环境设定为何呢！ ^_^

- 3. 服务器端的服务启动与埠口观察

启动实在太简单了，利用预设的 CentOS 启动方式来处理即可。

```
[root@www ~]# /etc/init.d/smb start  &lt;==这一版开始要启动两个daemon
[root@www ~]# /etc/init.d/nmb start
[root@www ~]# chkconfig smb on
[root@www ~]# chkconfig nmb on
[root@www ~]# netstat -tlunp &#124; grep mbd
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address         Foreign Address State   PID/Program name
tcp        0      0 :::139                :::*            LISTEN  1772/smbd
tcp        0      0 :::445                :::*            LISTEN  1772/smbd
udp        0      0 192.168.1.100:137     0.0.0.0:*               1780/nmbd
udp        0      0 192.168.100.254:137   0.0.0.0:*               1780/nmbd
udp        0      0 0.0.0.0:137           0.0.0.0:*               1780/nmbd
udp        0      0 192.168.1.100:138     0.0.0.0:*               1780/nmbd
udp        0      0 192.168.100.254:138   0.0.0.0:*               1780/nmbd
udp        0      0 0.0.0.0:138           0.0.0.0:*               1780/nmbd
```

特别注意，在 Samba 当中预设会启动多个端口，这包括数据传输的 TCP 端口 (139, 445)，以及进行 NetBIOS 名称解析之类工作的 UDP 埠口 (137, 138)，所以你才会看到很多数据的。那么能否仅支持 139 这个必要的埠口，关闭 445 呢？可以啊～透过 testparm -v 的观察，可以发现『 smb ports = 445 139 』这个设定值指定两个埠口的，因此你可以在 smb.conf 增加这个设定值，并改为 smb ports = 139 即可。不过，建议先保留默认值啦！

- 4. 假设自我为客户端的检验 (默认用 lo 接口)

关于客户端的观察我们会在后续进行介绍。在这里仅是说明如何确定我们的 Samba 设定与服务有顺利的在运作。 我们可以在本机上透过 smbclient 这支程序来处理，它的基本查询语法是这样的：

```
[root@www ~]# smbclient -L [//主机或IP] [-U 使用者账号]
选项与参数：
-L ：仅查阅后面接的主机所提供分享的目录资源；
-U ：以后面接的这个账号来尝试取得该主机的可使用资源
```

由于在这个范例当中我们并没有规范用户的安全等级 (share)，所以不必使用 -U 这个选项，因此你可以这样看看：

```
[root@www ~]# smbclient -L //127.0.0.1
Enter root's password: &lt;==因为不需要密码，因此这里单击 [Enter] 吧！
Domain=[VBIRDHOUSE] OS=[Unix] Server=[Samba 3.5.4-68.el6_0.2]

        Sharename       Type      Comment
        ---------       ----      -------
        temp            Disk      Temporary file space
        IPC$            IPC       IPC Service (This is vbird's samba server)
Domain=[VBIRDHOUSE] OS=[Unix] Server=[Samba 3.5.4-68.el6_0.2]

        Server               Comment
        ---------            -------
        VBIRDSERVER          This is vbird's samba server

        Workgroup            Master
        ---------            -------
        VBIRDHOUSE           VBIRDSERVER
```

上表输出的信息当中，分享的目录资源 (Sharename) 就是在 smb.conf 当中设定的 [temp] 名称啰！ 因此在这里的意思是：任何人都可以进入 //127.0.0.1/temp 这个目录当中， 而这个目录在 Linux 系统其实是 /tmp 目录。至于那个 IPC$ 则是为了要应付 Windows 环境所必须要存在的项目就是了。那么该如何使用这个资源呢？接下来我们可以利用 mount 这个指令来测试看看啰：

```
[root@www ~]# mount -t cifs //127.0.0.1/temp /mnt
Password: &lt;==因为没有密码，所以这里还是按 Enter 即可
[root@www ~]# df
Filesystem           1K-blocks      Used Available Use% Mounted on
....(前面省略)....
//127.0.0.1/temp/      1007896     53688    903008   6% /mnt

[root@www ~]# cd /mnt
[root@www mnt]# ll  &lt;==以上这两个动作要进行！才会知道有没有权限的问题！
[root@www mnt]# touch zzz
[root@www mnt]# ll zzz /tmp/zzz
-rw-r--r--. 1 nobody nobody 0 Jul 29 13:08 /tmp/zzz
-rw-r--r--. 1 nobody nobody 0 Jul 29 13:08 zzz
# 注意喔！你进入 /mnt 身份会被压缩成为 nobody 呢！不再是 root 啊！

[root@www mnt]# cd ; umount /mnt
```

确实可以挂载的起来，所以，测试完毕后，就将这个挂载的资料卸除吧。关于 mount 的用法，我们会在后面的小节继续介绍。

基本上，到此为止咱们就设定好一个简单的不需要密码即可登入的 Samba 服务器了！ 你可以先行到客户端软件功能的部分进行更细部的挂载测试。 接下来，让我们以简易的需要密码才能够登入 Samba 的方式来设计一个范例吧！

16.2.4 需账号密码才可登入的分享 (security = user)

设定一部不需密码即可登入的 Samba server 是非常简单的，不过， 你总不希望某些有机密性质的资料放在不设防的网芳中让大家查阅吧？ 举例来说，你总不希望你的家目录被人家随意浏览吧？家目录内可能有你自己的情书呢！^_^

那怎么办？没关系，我们可以透过 Samba 服务器提供的认证方式来进行用户权力的给予，也就是说，你在客户端联机到服务器时，必须要输入正确的账号与密码后，才能够登入 Samba 查阅到你自己的数据！ 那会不会很难啊？不会啦！ Samba 本身就提供一个小程序来帮助我们处理密码的建立了，整个流程还不太难。

比较重要的是 Samba 使用者账号必须要存在于 Linux 系统当中 (/etc/passwd)， 但是 Samba 的密码与 Unix 的密码档案并不相同 (这是因为 Linux 与网芳的密码验证方式及编码格式不同所致)。 这就比较有点小麻烦～没关系，就让我们依样画葫芦来处理一下这个部分的设定吧！

- 0. 假设条件

由于使用者层级会改变成 user 的阶段，因此 [temp] 已经没有必要存在！请将该设定删除或批注。 而服务器方面的整体数据则请保留，包括工作组等等的数据，并新增底下的资料：

- 使用者认证层级设定 (security) 为： user
- 用户密码档案使用 TDB 数据库格式，默认档案在 /var/lib/samba/private/ 内；
- 密码必须要加密；
- 每个可使用 samba 的使用者均拥有自己的家目录；
- 设定三个用户，名称为 smb1, smb2, smb3 ，且均加入 users 为次要群组。此三个用户 Linux 密码为 1234， Samba 密码则为 4321；
- 分享 /home/project 这个目录，且资源名称取名为： project；
- 加入 users 这个群组的使用者可以使用 //IP/project 资源，且在该目录下 users 这个群组的使用者具有写入的权限。

好了，开始一步步的处理吧！

- 1. 设定 smb.conf 配置文件与目录权限相关之设定

在这个范例的配置文件当中，我们会新增几个参数，新增的参数部分会用特殊字体圈起来，引用之前参数的部分则为一般字体。请交互参考看看啰：

```
# 1\. 开始设定重要的 smb.conf 档案呦！
[root@www ~]# vim /etc/samba/smb.conf
[global]
        workgroup       = vbirdhouse
        netbios name    = vbirdserver
        server string   = This is vbird's samba server
        unix charset    = utf8
        display charset = utf8
        dos charset     = cp950
        log file        = /var/log/samba/log.%m
        max log size    = 50
        load printers    = no

 # 与密码有关的设定项目，包括密码档案所在格式喔！
 security = user          &lt;==这行就是重点啦！改成 user 层级
        passdb backend = tdbsam  &lt;==使用的是 TDB 数据库格式！

# 2\. 分享的资源设定方面：删除 temp  加入 homes 与 project
[homes]                                  &lt;==分享的资源名称
        comment        = Home Directories
        browseable     = no              &lt;==除了使用者自己外，不可被其他人浏览
        writable       = yes             &lt;==挂载后可擦写此分享
        create mode    = 0664            &lt;==建立档案的权限为 664
        directory mode = 0775            &lt;==建立目录的权限为 775

[project]                                &lt;==就是那三位使用者的共享资源
        comment    = smbuser's project
        path       = /home/project       &lt;==实际的 Linux 上面的目录位置
        browseable = yes                 &lt;==可被其他人所浏览到资源名称(非内容)
        writable   = yes                 &lt;==可以被写入
        write list = @users              &lt;==写入者有哪些人的意思

# 2\. 每次改完 smb.conf 你都需要重新检查一下语法正确否！
[root@www ~]# testparm  &lt;==详细的 debug 请自行处理啰！
```

在上表当中比较有趣的设定项目主要有：

- [global] 修改与新增的部分：security 设定为 user 层级，且使用『passdb backend = tdbsam』这个数据库格式，因此密码文件会放置于 /var/lib/samba/private/ 内。 此外，默认密码就是加密的，因此不需要额外使用其他的设定参数来规范；

- [homes] 这个使用者资源共享部分： homes 是最特殊的资源共享名称，因为 Linux 上面的每位用户均有家目录，例如 smb1 的家目录位于 /home/smb1/ ，那当 smb1 用户使用 samba 时，她就会发现多了个 //127.0.0.1/smb1/ 的资源可用，而 smb2 就在 //127.0.0.1/smb2/ 这个资源。由于不可浏览 (browseable)，所以除了使用者可以看到自己的家目录资源外，其他人是无法浏览的。此外，为了规范权限，而多了 create mode 与 directory mode 两个设定值 (此值可设定也可不理会)；

- [project] 这个使用者资源共享部分：当我们新增一个共享资源时， 最重要的就是规范资源名称。在此例中我们使用 project 这个资源名称来指向 /home/project ，也就是说，//127.0.0.1/project 代表的是 /home/project 的意思。此外，能够使用这个资源的账号，为加入 users 这个群组的用户喔！ 透过 write list 这个项目比较单纯，如果是早期的设定，可能会使用 valid users ，但近来鸟哥比较偏好 write list 设定项目。 不过能否顺利的存取档案还与 Linux 最底层的档案权限有关。

千万不要忘记了，除了配置文件之外，详细的目录权限与账号设定等规范也要设定好！底下我们用范例来进行此项工作！

例题：我们预计要分享 /home/project 目录，这个目录的权限该如何设定？答：因为是要开放给 users 群组，而共享群组的权限通常是『 2770 』这个含有 SGID 的特殊旗标功能。因此这个目录应该如此设定才好：

```
[root@www ~]# mkdir /home/project
[root@www ~]# chgrp users /home/project
[root@www ~]# chmod 2770 /home/project
[root@www ~]# ll -d /home/project
drwxrws---. 2 root users 4096 Jul 29 13:17 /home/project
```

- 2. 设定可使用 Samba 的用户账号与密码

设定使用者账号是很重要的一环，因为设定错误的话，当然也就任何人都没有办法登入的！在这里我们必须先要说明一下 Linux 的文件系统与 SAMBA 设定的使用者登入权限的相关性！

- 在 Linux 这个系统下，任何程序都需要取得 UID 与 GID (User ID 与 Group ID) 的身份之后，才能够拥有该身份的权限，也才能够适当的进行存取档案等动作！

- 关于 Linux 这个系统的 UID 与 GID 与账号的相对关系，一般记录在 /etc/passwd 当中，当然也能透过 NIS, ldap 等方式来取对应；

- SAMBA 仅只是 Linux 底下的一套软件，使用 SAMBA 来进行 Linux 文件系统时，还是需要以 Linux 系统下的 UID 与 GID 为准则！

如果上面这几点说明你没有问题了，现在就来看一下当我们在 Windows 计算机上面以网络上的芳邻来连接 Linux 并且进行数据的存取时，会是怎样的一个情况呢？

我们需要透过 SAMBA 所提供的功能来进行 Linux 的存取，而 Linux 的存取是需要取得 Linux 系统上面的 UID 与 GID 的，因此，我们登入 SAMBA 服务器时，所利用 SAMBA 取得的其实是 Linux 系统里面的相关账号！这也就是说，在 **SAMBA** 上面的使用者账号，必须要是 **Linux** 账号中的一个！

所以说，在不考虑 NIS 或 LDAP 等其他账号的验证方式，单纯以 Linux 本机账号 (/etc/passwd) 作为身份验证时， 在 Samba 服务器所提供可登入的账号名称，必须要存在于 /etc/passwd 当中！ 这是一个很重要的概念！例如你要先有 dmtsai 在 /etc/passwd 当中后，才能将 dmtsai 加入 Samba 的使用者当中。这都是很基本的账号权限概念，如果你觉得这里阅读方面有问题， 若不考虑鸟哥的解释不良，表示你必须要回去读读基础篇了～ ^_^

现在我们知道需要新增 smb1, smb2, smb3 三个用户，且这三个用户需要加入 users 群组。此外，我们之前还建立过 student 这个用户，假设这四个人都需要能用 Samba 服务，那么除了新增用户之外，我们还需要利用 pdbedit 这个指令来处理 Samba 用户功能喔！

```
# 1\. 先来建立所需要的各个账号，但假设 student 已经存在了喔！
[root@www ~]# useradd -G users smb1
[root@www ~]# useradd -G users smb2
[root@www ~]# useradd -G users smb3
[root@www ~]# echo 1234 &#124; passwd --stdin smb1
[root@www ~]# echo 1234 &#124; passwd --stdin smb2
[root@www ~]# echo 1234 &#124; passwd --stdin smb3

# 2\. 使用 pdbedit 指令功能
[root@www ~]# pdbedit -L [-vw]            &lt;==单纯的察看帐户信息
[root@www ~]# pdbedit -a&#124;-r&#124;-x -u 账号     &lt;==新增/修改/删除账号
[root@www ~]# pdbedit -a -m -u 机器账号    &lt;==与 PDC 有关的机器码
选项与参数：
-L ：列出目前在数据库当中的账号与 UID 等相关信息；
-v ：需要搭配 -L 来执行，可列出更多的讯息，包括家目录等数据；
-w ：需要搭配 -L 来执行，使用旧版的 smbpasswd 格式来显示数据；
-a ：新增一个可使用 Samba 的账号，后面的账号需要在 /etc/passwd 内存在者；
-r ：修改一个账号的相关信息，需搭配很多特殊参数，请 man pdbedit；
-x ：删除一个可使用 Samba 的账号，可先用 -L 找到账号后再删除；
-m ：后面接的是机器的代码 (machine account)，与 domain model 有关！

# 2.1 开始新增使用者吧！
[root@www ~]# pdbedit -a -u smb1
new password: &lt;==输入 4321 这个密码瞧瞧
retype new password: &lt;==再输入一次吧！
Unix username:        smb1   &lt;==底下为输入正确后的显示结果！
NT username:
Account Flags:        [U          ]
User SID:             S-1-5-21-4073076488-3046109240-798551845-1000
Primary Group SID:    S-1-5-21-4073076488-3046109240-798551845-513
Full Name:
Home Directory:       \\vbirdserver\smb1
HomeDir Drive:
Logon Script:
Profile Path:         \\vbirdserver\smb1\profile
Domain:               VBIRDSERVER
Account desc:
Workstations:
Munged dial:
Logon time:           0
Logoff time:          9223372036854775807 seconds since the Epoch
Kickoff time:         9223372036854775807 seconds since the Epoch
```

```
Password last set:    Fri, 29 Jul 2011 13:19:56 CST
Password can change:  Fri, 29 Jul 2011 13:19:56 CST
Password must change: never
Last bad password   : 0
Bad password count  : 0
Logon hours         : FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
# 你可以发现其实讯息非常的多！若需修改细部设定，请 man pdbedit 吧！
[root@www ~]# pdbedit -a -u smb2
[root@www ~]# pdbedit -a -u smb3
[root@www ~]# pdbedit -a -u student

# 2.2 查询目前已经存在的 Samba 账号
[root@www ~]# pdbedit -L
smb1:2004:
smb3:2006:
smb2:2005:
student:505:
# 仅会列出账号与 UID 而已呦！

# 2.3 尝试修改与删除 smb3 这个账号看看
[root@www ~]# smbpasswd smb3
New SMB password:
Retype new SMB password:
# 修改密码比较特殊，管理密码参数是使用 pdbedit，修改密码得要用 smbpasswd 呦！

[root@www ~]# pdbedit -x -u smb3
[root@www ~]# pdbedit -Lw
# 此时你就看不到 smb3 这个用户啰！所以测试完请立即将它加回来！
```

以后如果有需要新增额外的使用者账号，若该账号原本不存在，则使用 useradd 再以 pdbedit -a 去新增。 若已经存在于 Linux 的实体账号，直接用 pdbedit -a 新增即可。同时要注意，管理 TDB 数据库格式建议使用 pdbedit 这个新的玩意儿来处理，smbpasswd 仅剩下修改密码的功能需记忆即可！

- 3. 重新启动 Samba 并进行自我测试

在经过重新启动后，我们所进行的修订才会生效。然后使用 smbclient 来检查看看，是否不同身份会有不一样的浏览结果呢？ 赶紧看看：

```
[root@www ~]# /etc/init.d/smb restart
[root@www ~]# /etc/init.d/nmb restart

# 1\. 先用匿名登录试看看！
[root@www ~]# smbclient -L //127.0.0.1
Enter root's password:      &lt;==直接按下 [Enter] 即可。
Anonymous login successful  &lt;==有看到匿名的字样了！
Domain=[VBIRDHOUSE] OS=[Unix] Server=[Samba 3.5.4-68.el6_0.2]

        Sharename       Type      Comment
        ---------       ----      -------
        project         Disk      smbuser's project
        IPC$            IPC       IPC Service (This is vbird's samba server)
....(底下省略)....

# 2\. 再使用 smb1 这个账号登入试看看！
[root@www ~]# smbclient -L //127.0.0.1 -U smb1
Enter smb1's password:  &lt;==输入 smb1 在 pdbedit 所建立的密码！
Domain=[VBIRDHOUSE] OS=[Unix] Server=[Samba 3.5.4-68.el6_0.2]

        Sharename       Type      Comment
        ---------       ----      -------
        project         Disk      smbuser's project
        IPC$            IPC       IPC Service (This is vbird's samba server)
        smb1            Disk      Home Directories &lt;==多了这玩意儿！
....(底下省略)....
```

由上表我们可以发现，经由不同的身份登入可以取得不一样的浏览数据， 所以在使用上面需要特别留意喔！接下来，让我们开始来自我挂载测试看看！

```
[root@www ~]# mount -t cifs //127.0.0.1/smb1 /mnt -o username=smb1
Password: &lt;==确定是输入正确的密码喔！
# 此时 /home/smb1/ 与 /mnt 应该拥有相同的档名才对！因为挂载嘛！

[root@www ~]# ll /home/smb1/.bashrc
-rw-r--r--. 1 smb1 smb1 124 May 30 23:46 /home/smb1/.bashrc &lt;==确定有档案
[root@www ~]# ls -a /mnt
# 却看不到任何东西！应该是 SELinux 的问题吧！根据 /var/log/messages 的讯息，
# 进行如下的动作就能够处理好这个程序！

[root@www ~]# setsebool -P samba_enable_home_dirs=1
[root@www ~]# ls -a /mnt
.  ..  .bash_logout  .bash_profile  .bashrc  .gnome2  .mozilla
# 档名出现啦！OKOK！这个使用者挂载处理完毕！

[root@www ~]# umount /mnt
```

自我测试是非常重要的！因为 Samba 是会对外提供服务的，因此 SELinux 会特别『关照』一下这个服务！包括默认用户家目录不会有开放的权限、预设的 SELinux type 不对就无法使用 (你可以自己尝试挂载 //127.0.0.1/project 就知道啥原因啰！)，所以，自行测试完毕就能够理解哪个地方的 SELinux 没有设定妥当！详细的设定请到 16.2.6 安全性设定去查阅。

Tips: 根据网友回报，因为之前我们设定的 security 是 share，而且已经使用 Windows 系统测试过，在同一部 Windows 系统上面重复测试时，会发生无法登入的情况。建议直接将 windows 系统重新启动清除前一次登入的信息即可！ ^_^



- 4. 关于权限的再说明与累加其他分享资源的方式：

有的时候你会发现，明明在 smb.conf 当中已经设定了 writable 可写入，使用者登入的身份也没有问题， 为啥就是无法挂载或写入呢？是否是服务器设定哪里还有问题啊？非也非也！主要的问题常常是来自于 Linux 文件系统的权限啦！

举上面的例子来说，当你无法挂载却发现 Linux 传统权限是对的，那么肯定是 SELinux 出问题～这部份得要用 setsebool 与 chcon 或 restorecon 等指令来克服。另外就是，我们在 smb.conf 当中设定 [project] 为可写入，亦即 /home/project 是可写入的。假设 smb1 属于 users 这个群组，因此以 smb1 登入 SAMBA 服务器后，对于 /home/project 应该是具有可以读写的能力的！但是，如果你以 root 的身份建立 /home/project 却又忘记修改权限的话， 此时 /home/project 是无法让 users 这个群组写入的，因此 smb1 这个使用者当然不具有写入的能力。 这样说，了解鸟哥想要说啥了吗？注意注意喔！ ^_^

那如果你还要扩充分享的目录与能够登入的使用者时，可以这样做：

- 利用编辑 smb.conf 来开放其他的目录资源，并且特别注意 Linux 在该目录下的权限喔！请使用 chown 与 chmod 吧！
- 利用 pdbedit 来新增其他可用 Samba 的账号，如果该账号并没有出现在 /etc/passwd 里面，请先以 useradd 新增该账号；
- 不论进行完任何的设定，请先以 testparm 进行确认，之后以 /etc/init.d/{smb,nmb} restart 来重新启动！

事实上，SAMBA 的一般用途就是在这个联机的模式中！多使用 SAMBA 来分享你的资源吧！鸟哥都是使用 SAMBA 来做为远程服务器与我的工作机互通有无的重要媒介说～

16.2.5 设定成为打印机服务器 (CUPS 系统)

时至今日，打印机的网络功能已经很强悍了！甚至也有支持无线网络的打印机，因此每台打印机都可以独立作为各个 PC 的独自的打印机，老实说也没有必要进行 Samba 的网络打印机服务器啦！但毕竟还是有些比较旧型的机种， 或者买不起有内建网络的打印机时，那么 Samba 的打印机服务器还是有存在的价值啰。

在 Linux 底下进行打印的服务很多，不过我们这里要介绍的仅有目前较广为流行的 CUPS (Common Unix Printing System) 这一个。 详细的 CUPS 安装设定方法我们已经在基础篇第三版第二十一章 CUPS 当中提过，所以这里我们不再详细说明，仅介绍大致的处理流程就是了。如果你需要较早期的 LPRng 打印系统的话， 建议可以参考底下的数据喔：

- 依玛猫的打印文件：http://www.imacat.idv.tw/tech/lnxprint.html
- 鸟哥的 LPRng 简介：http://linux.vbird.org/linux_server/0370samba/0370samba.php

Tips: 在这个小节中，鸟哥假设你的打印机并不是网络打印机，而是使用 USB 接口连接的打印机格式。如果你的打印机真的有支持网络， 那建议直接参考打印机手册来设定即可，不需要安装 Samba 打印机。因为某些厂牌的打印机网络卡有特殊的功能， 例如 HP 的网卡通常还支持某些特殊的打印功能 (双面、多页打印等)，这些功能透过伺服器重新分享时，可能会遗失！



- 0. 假设条件

既然要分享打印机，就得要有打印机啊！鸟哥使用对 Linux 支持度较高的 HP LaserJet P2015dn 这部打印机为例， 不使用网络功能，单纯使用 USB 连接到 Samba 服务器上。

- CUPS 连接到 USB 打印机，并且开放非本机的 IP 来源使用此打印机；
- 使用 CUPS 内建的打印机驱动程序；
- 前往 HP 打印机官网取得 Windows 操作系统的驱动程序；

- 1. 安装打印机与确定打印机的联机正常

再次说明，并不是所有的打印机都被 Linux 所支持的，所以当你想要链接一部打印机到 Linux 系统上头时， 请务必到 http://www.openprinting.org/printers 上头去看看是否有被支持喔！ 如果没有被支持，那就换一部打印机吧！不要进行垂死的挣扎了...

如果你的打印机端口为使用 USB 或者是平行串行端口的话，那么当你连接上打印机后， 可以利用底下的方式测试看看是否成功的连接上了：

```
[root@www ~]# lsusb
Bus 001 Device 002: ID 03f0:3817 Hewlett-Packard LaserJet P2015 series
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
[root@www ~]# ll /dev/usb/lp0
crw-rw----. 1 root lp 180, 0 Jul 29 13:55 /dev/usb/lp0
# 看得出来，已经有个 lp0 的打印机啰！测试打印一下吧！

[root@www ~]# echo "Hello printer" &gt; /dev/usb/lp0
```

如果打印机有响应，这表示 OK 的啦！你可以进行底下的工作了。

- 2. 设定 CUPS 与打印机的联机

预设 CUPS 都会开启，不过，因为我们安装的是『 basic server 』的模式，所以 CUPS 默认并没有被安装起来。 所以这里要安装且重新设定与启动才行。本章节 CUPS 的设定原则是这样的：

- 我需要让 192.168.100.0/24 这个网域可以使用打印机
- 我需要让 192.168.100.0/24 及 127.0.0.0/8 可以管理 CUPS 系统

然后开始这样做：

```
[root@www ~]# yum groupinstall "Print Server"
[root@www ~]# vim /etc/cups/cupsd.conf
# 1\. 让监听的接口开放在所有接口！
# Listen localhost:631  &lt;==约在第 18 行左右，改成如下：
Listen 0.0.0.0:631

# 2\. 让内部网域能够进行 CUPS 的浏览与管控
&lt;Location /&gt;        &lt;==约在 32 行左右，新增能够让内网其他 IP 浏览者
  Order allow,deny
 Allow From 127.0.0.0/8
  Allow From 192.168.100.0/24
&lt;/Location&gt;

&lt;Location /admin&gt;       &lt;==约在 39 行左右，新增能够管理 CUPS 者
  Encryption Required   &lt;==因为这里的关系，所以可能会用 https://IP 喔！
  Order allow,deny
 Allow From 127.0.0.0/8
  Allow From 192.168.100.0/24
&lt;/Location&gt;
```

设定完毕后就可以开始来启动 cups 系统，可以这样做：

```
[root@www ~]# /etc/init.d/cups start
[root@www ~]# chkconfig cups on
[root@www ~]# netstat -tunlp &#124; grep 'cups'
tcp      0  0  0.0.0.0:631         0.0.0.0:*      LISTEN      1851/cupsd
udp      0  0  0.0.0.0:631         0.0.0.0:*                  1851/cupsd
```

那个 631 的埠口就是 CUPS 所启动的啦！要注意的是，开放界面得要给 0.0.0.0 才对哟！然后我们可以开始设定打印机了！ 由于 CUPS 支持很多不同的打印机端口，每种端口都不一样，常见的有：

- USB 端口： usb:/dev/usb/lp0
- 网络打印机： ipp://ip/打印机型号
- 网络芳邻打印机： smb://user:password@host/printer

之所以要加上 192.168.100.0/24 可以控制服务器 CUPS 的原因在于...鸟哥的服务器没有 X 窗口啦！ 所以需要透过平时的工作机连上服务器才行啊！此时，将 CUPS 开放在区网内可以控制的功能就很重要啦！ 此外，因为鸟哥的主机所在环境问题，这部 192.168.100.254 还有一个界面为 192.168.1.100，鸟哥在 cupsd.conf 里面也加入这个网段了 (上面的范例中并没有特别强调)，所以底下的图示你会看到很多 192.168.1.100 的 IP ，不要害怕！那是正常的！^_^！好了，请打开浏览器，在网址列输入： https://192.168.100.254:631 (底下则是 192.168.1.100)

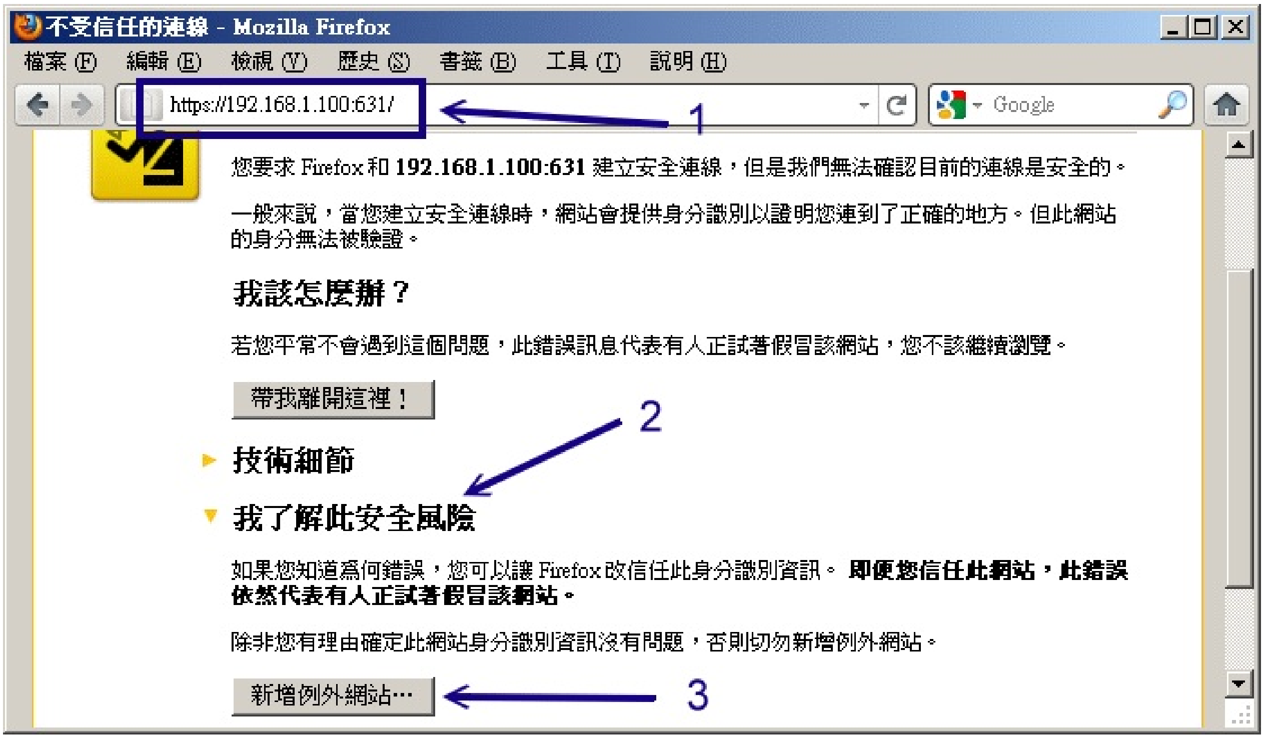


图 16.2-1、用 CUPS 设定 USB 打印机

如上图所示，由于我们使用的是 https 这个需要凭证的联机模式，因此就会出现这个不受信任的网站讯息。 没关系，你直接按下『我了解安全风险』后，再选择『新增例外网站』即可出现如下图示：

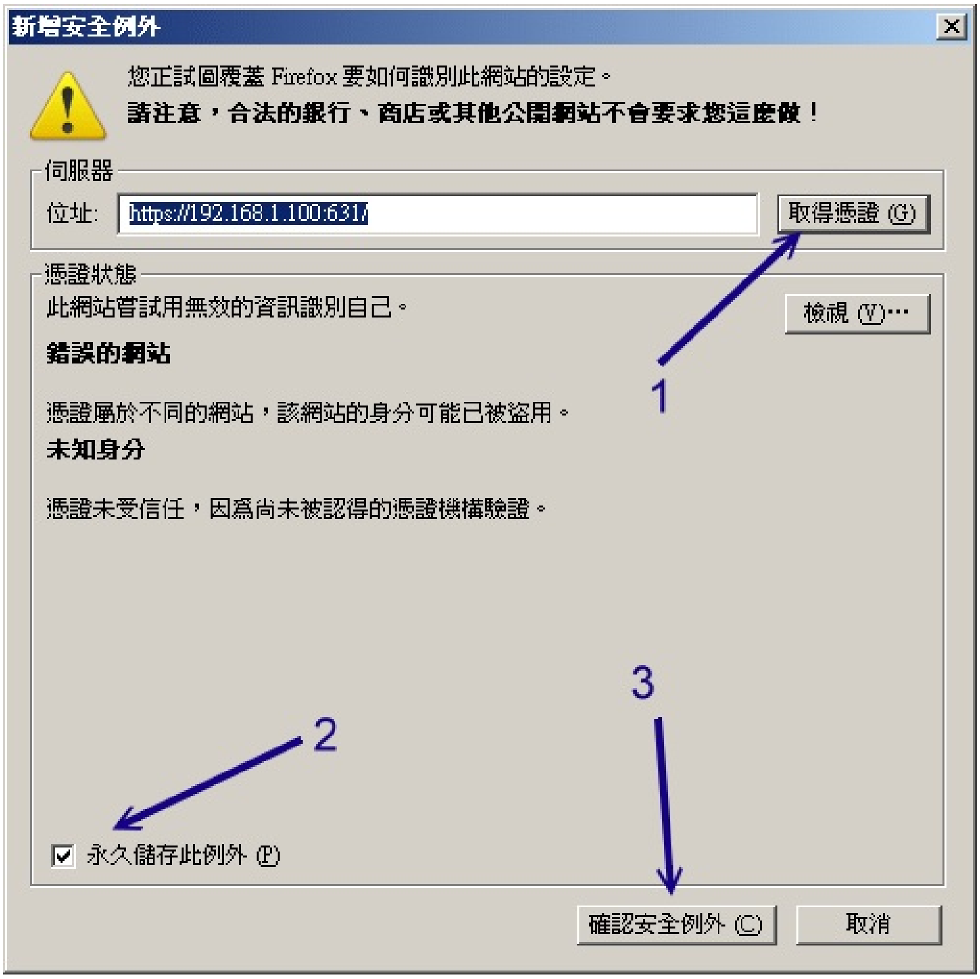


图 16.2-2、用 CUPS 设定 USB 打印机

如果这部主机真的是你的，那么就选择箭头 2 所指的那个『永久储存』吧！最后按下箭头 3 所指的『确认安全例外』即可！如果一切顺利，就会出现如下的 CUPS 设定图示：

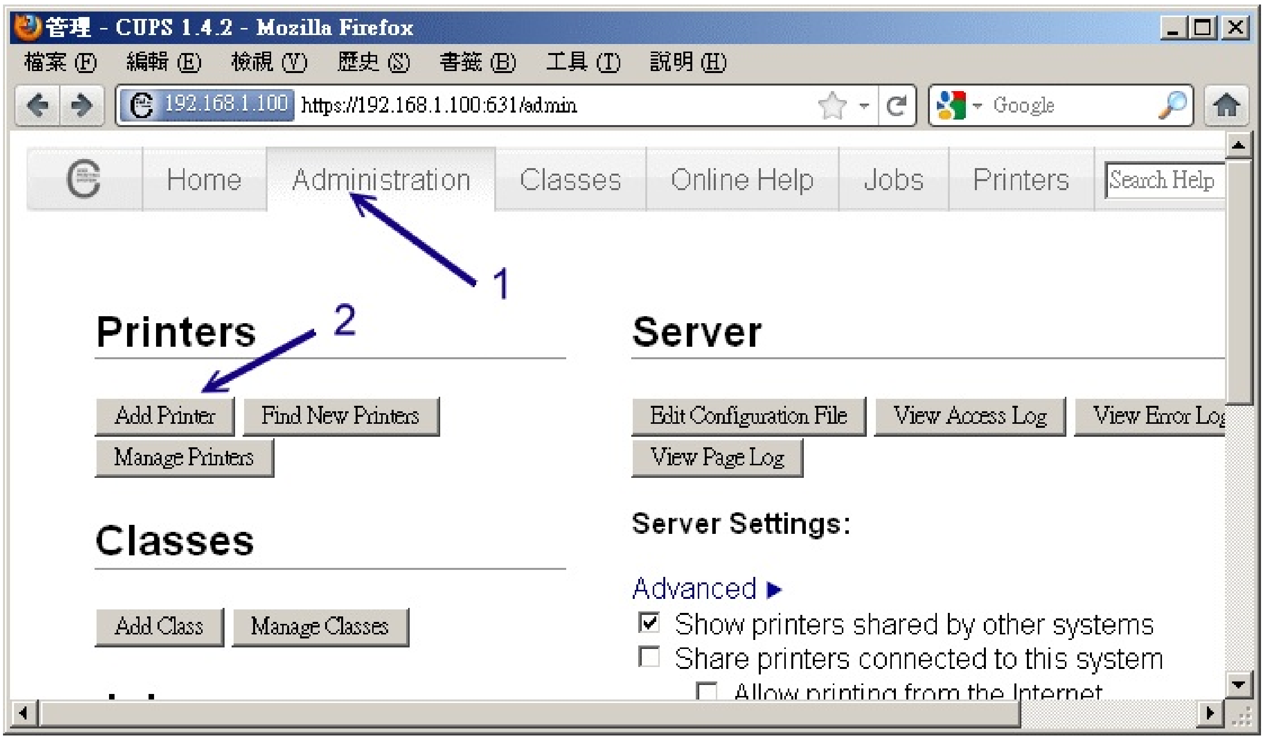


图 16.2-3、用 CUPS 设定 USB 打印机

在上头的欢迎图示当中，由于我们是想要建立打印机，因此点选箭头(1)所指的那个按钮进入打印机功能，然后点选 (2)来建立打印机吧！



图 16.2-4、用 CUPS 设定 USB 打印机

这一版比较有趣的地方，是会先让你输入账号与密码才进行后续的动作哩！所以这里请输入 root 的帐密吧！

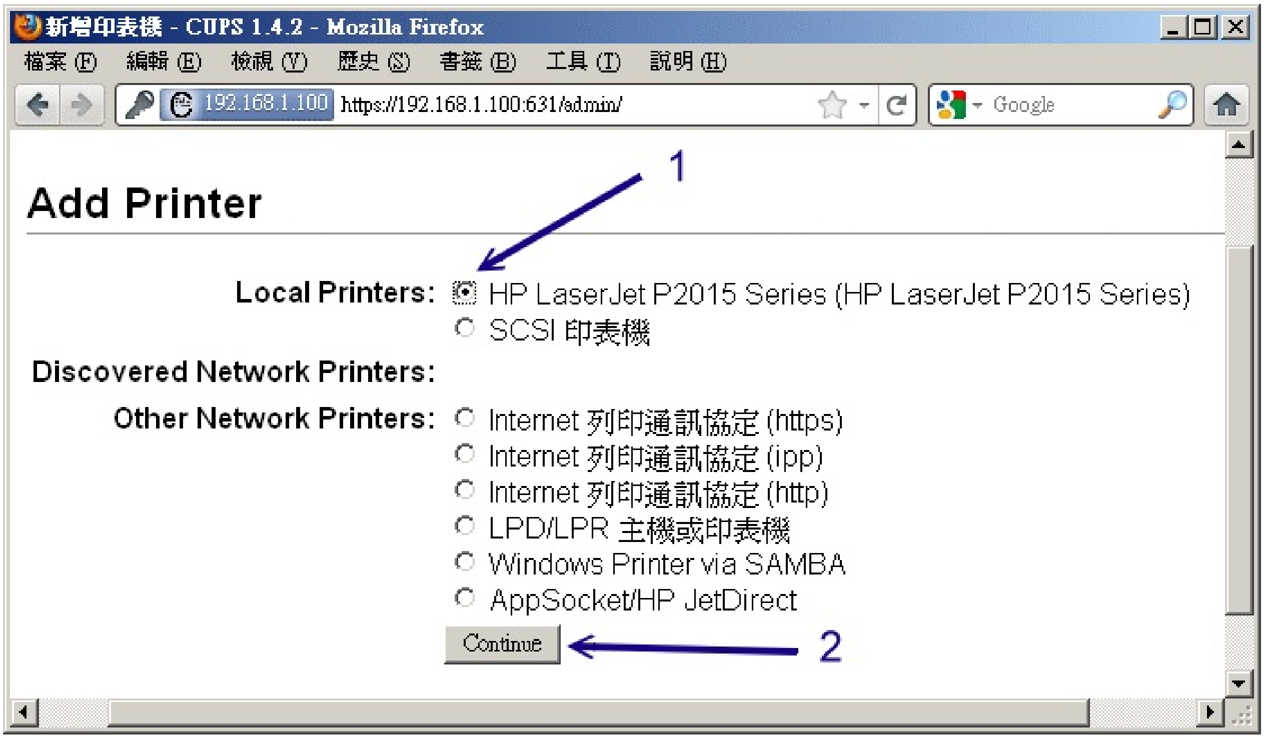


图 16.2-5、用 CUPS 设定 USB 打印机

在上面的图示中，你应该要选择的是我们这部本机的 USB 打印机装置才对。该装置是由 HAL 服务所自动侦测到的， 如果你没有看到任何 USB 的打印机，那可能就得要查询一下打印机电源是否正确的开启了！ 点选他吧！

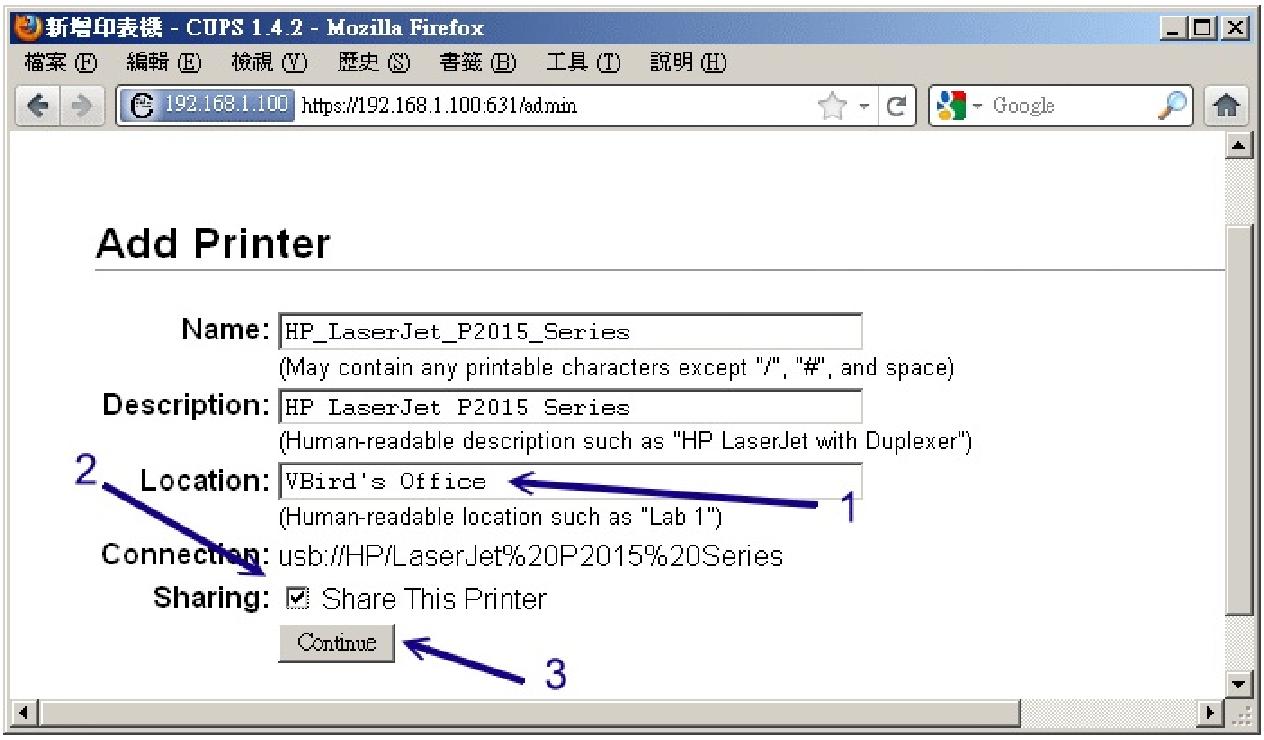


图 16.2-6、用 CUPS 设定 USB 打印机

建立打印机时，最重要的是那个打印队列 (上面方框中的第一个，名称的那个玩意儿)，在这里鸟哥使用 CUPS 预设帮我捉到的档名。 这个名称很重要，是未来分享出的打印机名字啰！ 至于位置与描述就随便你填啰。由于我们是想要做成打印服务器，所以『share this printer 』

当然要勾选！当你按下『继续』后，就会出现如下图示：

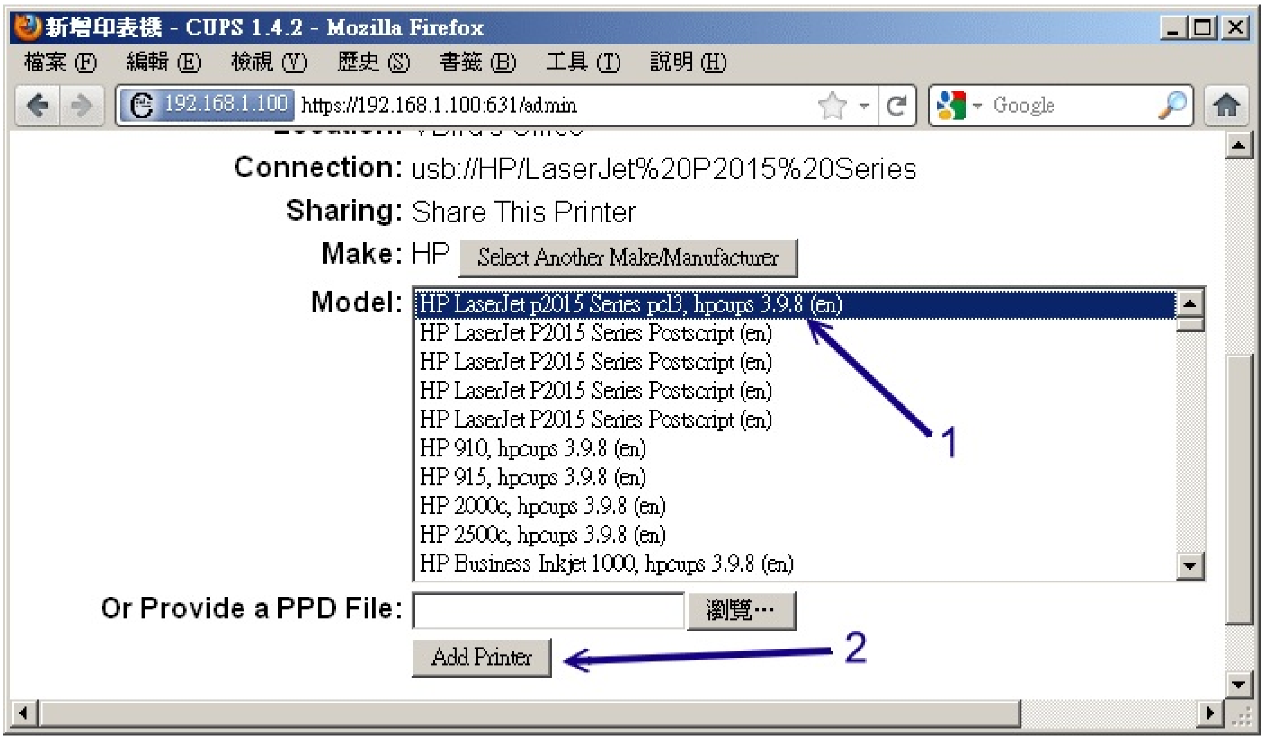


图 16.2-7、用 CUPS 设定 USB 打印机

接下来 CUPS 会帮你选择一个相对较佳的驱动程序，基本上，使用 CUPS 帮你捉到的预设驱动程序应该就 OK 了！ 选完后请按下『加入打印机』按钮吧！

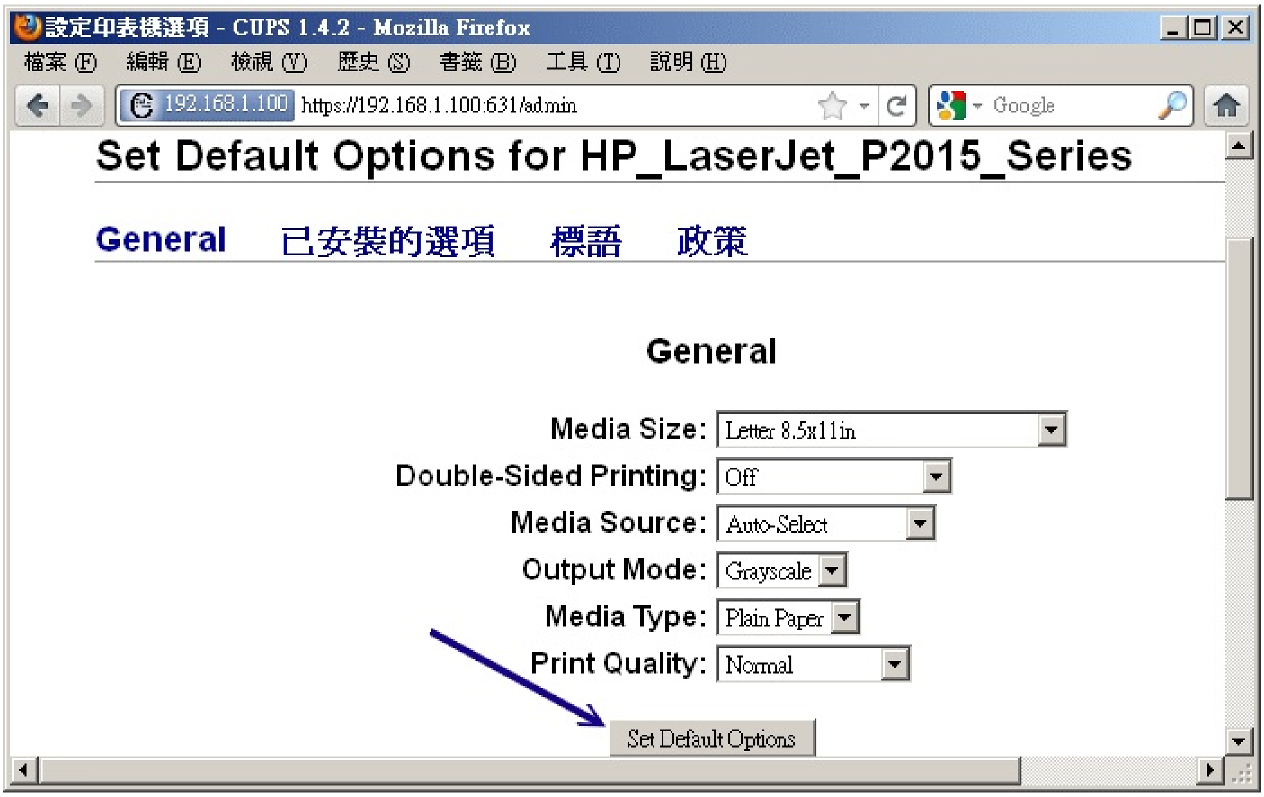


图 16.2-8、用 CUPS 设定 USB 打印机

看你还有没有要修改其他的预设参数，如果没有的话，就按下图 16.2-8 的『Set Default Options』按钮吧！ 如果一切没有问题，你的打印机就设定妥当了。如果想要查阅打印机的详细信息，那可以点选 Printer 的项目！如下图所示：

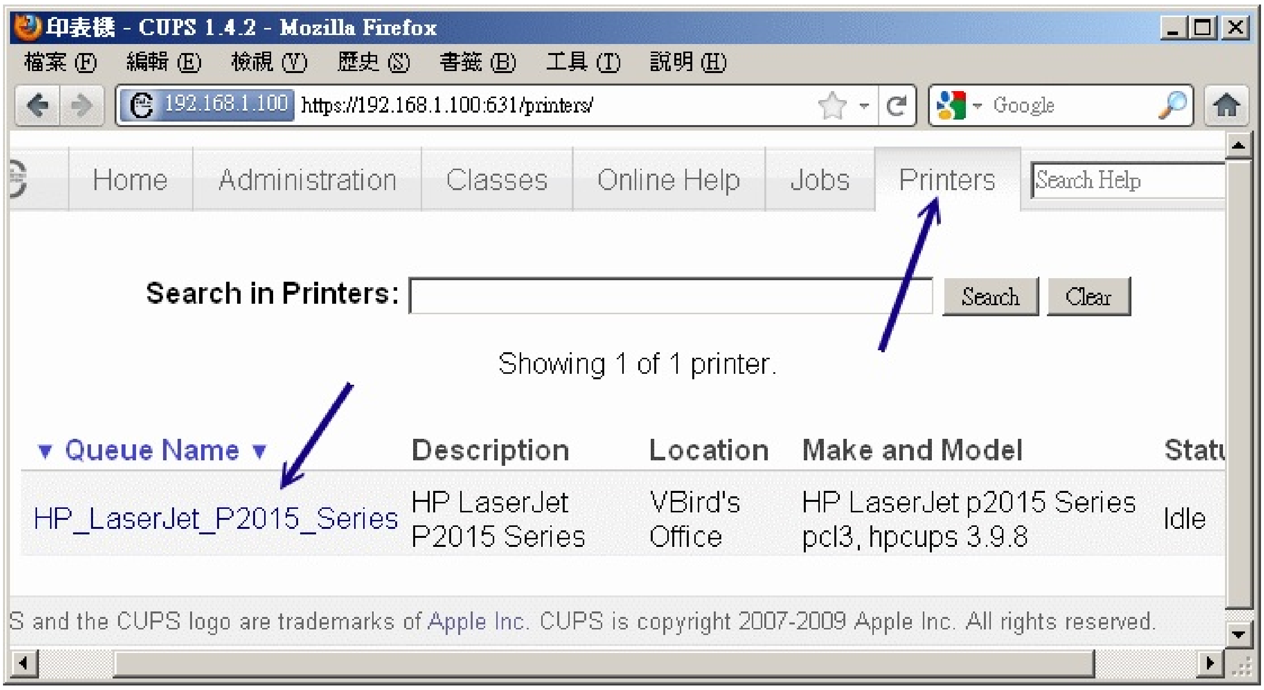


图 16.2-9、用 CUPS 设定 USB 打印机

如果都正常没问题，那么你的系统已经有一部打印机被 CUPS 所管理，且这部打印机在网络的网址为：

- http://服务器IP:631/printers/打印机队列名称
- http://192.168.100.254:631/printers/HP_LaserJet_P2015_Series

接下来看看如何将它连结到咱们的 Samba 服务器中吧！

- 3. 在 smb.conf 当中加入打印机的支持 (Optional)

开始告诉 Samba 将这部打印机给他分享出去吧！你需要这样处理：

```
[root@www ~]# vim /etc/samba/smb.conf
[global]
        # 得要修改 load printers 的设定，然后新增几个数据
        load printers = yes
        cups options  = raw        &lt;==可支持来自 Windows 用户的打印作业
        printcap name = cups
        printing      = cups       &lt;==与上面这两个在告知使用 CUPS 打印系统

[printers]                         &lt;==打印机一定要写 printers 喔！
        comment = All Printers
        path    = /var/spool/samba&lt;==预设把来自 samba 的打印作业暂时放置的队列
        browseable = no            &lt;==不被外人所浏览啦！有权限才可浏览
        guest ok   = no            &lt;==与底下两个都不许访客来源与写入(非文件系统)
        writable   = no
        printable  = yes           &lt;==允许打印很重要的一项工作！

[root@www ~]# testparm  &lt;==若有错误，请自行处理一下
[root@www ~]# /etc/init.d/smb restart
[root@www ~]# /etc/init.d/nmb restart
```

基本上透过这样的设定你的 Samba 就能够顺利的提供打印机的服务了！不过可惜的是，Windows 客户端依旧得要安装打印机的驱动程序才能够使用 Samba 所提供的打印机，此时真是麻烦兼讨厌啊～有没有可能让 Samba 主动的提供驱动程序给使用者，这样一来客户端就不需要额外去找驱动程序啰！是可以的，透过 Samba 3.x 即可处理！就这么巧， CentOS 的 Samba 就是 3.x 呢！所以我们可以透过底下的方式来处理。

- 4. 让 Samba 主动提供驱动程序给 Windows 用户使用

另外，或许你会想，打印机的型号这么多，那么 Linux 该如何提供这些打印机的驱动程序啊？岂不麻烦？ 还好啦， CUPS 主要是透过利用 Postscript 的打印语言与打印机沟通的，因此客户端只要取得 postscript 的驱动程序他们就能够使用咱们的 Samba 服务器所提供的打印机了！如此一来， 不论打印机的型号为何，只要他们能够支持 Postscript 的打印格式，OK 搞定！而且 CUPS 官网本身就有提供 CUPS 的 Postscript 驱动程序啰！可以到底下的连结去下载：

- 支持 CUPS 的软件：http://www.cups.org/software.php

很棒的是，因为我们是 CentOS 6.x 有支持 rpm 软件封装的系统，因此可以直接下载 cups-windows-6.0-1.i386.rpm 这个档案即可，直接安装这个 rpm 档案就能够取得 cups 对 Windows 的打印机驱动程序了。 这个档案安装完毕之后，会将驱动程序放置于 /usr/share/cups/drivers/ 里头呦！ 不过你得要注意的是，除了这个驱动程序外，要支持 Windows 2000 以后出产的 Windows 版本， 你还得到 Windows XP 底下的目录去下载几个 32 位支持的档案：

- Win XP 32 位：C:\WINDOWS\system32\spool\drivers\w32x86\3

将该目录下里面的 PS 开头的档案通通下载下来，应该有四个档案的，请将他复制成为档名小写的档案，并且放置到你 Samba 服务器上的 /usr/share/cups/drivers/ 目录下，该目录内放置的这就是基本的驱动程序说！ 在鸟哥的这个目录底下至少含有这几个档案就是了：

```
[root@www ~]# ll /usr/share/cups/drivers
-rw-r--r-- 1 root root    803  4月 20  2006 cups6.inf
-rw-r--r-- 1 root root     72  4月 20  2006 cups6.ini
-rw-r--r-- 1 root root  12568  4月 20  2006 cupsps6.dll
-rw-r--r-- 1 root root  13672  4月 20  2006 cupsui6.dll  &lt;==上面为 cups 提供
-rw-r--r-- 1 root root 129024  3月 24 13:29 ps5ui.dll    &lt;==底下为 Win XP 提供
-rw-r--r-- 1 root root 455168  3月 24 13:29 pscript5.dll
-rw-r--r-- 1 root root  27568  3月 24 13:29 pscript.hlp
-rw-r--r-- 1 root root 792644  3月 24 13:29 pscript.ntf
```

上述的档案鸟哥将他打包成为一个档案了，你可以在底下的连结下载：

- http://linux.vbird.org/linux_server/0370samba/cups-samba-windowsxp.tgz

不过你得注意，这个档案内的 Windows 数据是由 32 位的 Windows XP 上面捉来的，所以对于 Windows 98/ME 是没有作用的。同时，对于 64 位的其他较晚期的 Windows 7 等系统可能就得要重新处理了！ 你得自行上网查阅相关的数据下载方式喔。接下来我们必须要在

smb.conf 里面增加一笔新的分享数据，这个分享数据必须是 [print$] 名称才行！有点类似这样啦：

```
[root@www ~]# vim /etc/samba/smb.conf
[global]
....(设定保留原本数据)....
[homes]
....(设定保留原本数据)....
[printers]
....(设定保留原本数据)....
[print$]
        comment    = Printer drivers
        path       = /etc/samba/drivers  &lt;==存放打印机驱动程序的目录
        browseable = yes
        guest ok   = no
        read only  = yes
        write list = root                &lt;==这个驱动程序的管理员
[project]
....(设定保留原本数据)....

[root@www ~]# mkdir /etc/samba/drivers
[root@www ~]# chcon -t samba_share_t /etc/samba/drivers
# 由于预设的 CUPS 仅有 root 能管理，因此我们以 root 作为打印机管理员；
# 同时 SELinux 的类型也要修订如上的方式！那 root 就得要加入 samba 的支持才行：
[root@www ~]# pdbedit -a -u root

[root@www ~]# testparm                 &lt;==测试语法
[root@www ~]# /etc/init.d/smb restart  &lt;==重新启动

[root@www ~]# smbclient -L //127.0.0.1 -U root
Enter root's password:  &lt;==输入 root 在 samba 的密码先
Domain=[VBIRDHOUSE] OS=[Unix] Server=[Samba 3.5.4-68.el6_0.2]

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer drivers
        project         Disk      smbuser's project
        HP_LaserJet_P2015_Series Printer    HP LaserJet P2015 Series
        IPC$            IPC       IPC Service (This is vbird's samba server)
        root            Disk      Home Directories
# 瞧！有看到一部打印机以及驱动程序所在的分享数据啰！
```

现在我们要告知 Samba 说，我们的 CUPS 可提供 Windows 客户端的驱动程序，所以用户不需要自行设定他们的驱动程序哩！ 要由 cups 告知 Samba 是由 cupsaddsmb 这个指令来搞定的，整个指令的执行很简单的：

```
[root@www ~]# cupsaddsmb [-H SAMBA 服务器名] [-h CUPS 服务器名] \
&gt;   -a -v [-U 使用者账号]
选项与参数：
-H ：后续接的是 Samba 服务器名，本机的话可以直接用 localhost 即可；
-h ：后续接的为 CUPS 的服务器名，同样的可使用 localhost 即可；
-a ：自动搜寻出所有可用的 CUPS 打印机；
-v ：列出更多的信息；
-U ：打印机管理员

# 利用前面的说明将打印机驱动程序挂上 SAMBA (注意 CUPS 管理员预设是 root)
[root@www ~]# cupsaddsmb -H localhost -U root -a -v
Password for root required to access localhost via SAMBA: &lt;==root 在 SAMBA 密码
# 这里会闪过很多的讯息，说明已经安装了某些信息，底下鸟哥仅列出简单的讯息而已。
Running command: smbclient //localhost/print$ -N -A /tmp/cupsbrdBaE -c 'mkdir
W32X86;put /tmp/cupsu13OSU W32X86/HP_LaserJet_P2015_Series.ppd;...

[root@www ~]# ll /etc/samba/drivers
drwxr-xr-x. 3 root root 4096 Jul 29 15:15 W32X86  &lt;==这就是驱动程序目录
```

最后在驱动程序的存放目录会多出一个 W32X86 的目录，你可以查询一下该目录的内容， 那就是预计要给客户端使用的驱动程序啦！这样就搞定了！不过，为了将所有的数据通通驱动， 建议你将 CUPS 及 SAMBA 通通重新启动吧！

```
[root@www ~]# /etc/init.d/cups restart
[root@www ~]# /etc/init.d/smb restart
[root@www ~]# /etc/init.d/nmb restart
```

- 5. 一些问题的克服：

如果一切顺利的话，你在 Windows 客户端应该可以顺利的连接到打印机啰！ 开心吧！不过，如果你曾经印错数据，那么该如何进入 Linux 的 Samba 主机将该数据移除呢？ 你最好知道底下的几个指令，关于这些指令的进阶用法则请自行给他 man 看看了：

```
# 1\. 列出所有可用的打印机状态
[root@www ~]# lpstat -a
HP_LaserJet_P2015_Series accepting requests since Fri 29 Jul 2011 02:55:28 PM CST

# 2\. 查询目前默认打印机的的工作情况
[root@www ~]# lpq
hpljp2015dn 已就绪
没有项目
# 列出打印机的工作，若有打印作业存在时 (例如关掉打印机再印测试页)，会如下所示：
hpljp2015dn 已就绪并正在打印
等级     拥有人   工作     档案                          总计   大小
active  root     2       Test Page                       17408 byte

# 3\. 删除所有的工作项目喔！
[root@www ~]# lprm -
# 加上那个减号 (-) 代表移除所有等待中的打印作业！
```

打印作业就是这样进行的啦！赶紧试看看吧！接下来探讨一下相关的防火墙与安全性的讨论！

16.2.6 安全性的议题与管理

使用 SAMBA 其实是有一定程度的危险性的，这是因为很多网络攻击的蠕虫、病毒、木马就是透过网芳来攻击的！ 为了抵挡不必要的联机，所以 CentOS 5.x 预设的 SELinux 已经关闭了很多 Samba 联机的功能， 因此预设情况下，很多客户端的挂载可能会有问题。此外，仅开放有权限的网域来源，以及透过 smb.conf 来管理特定的权限，也是很重要的！同时，Linux 文件系统的 r, w, x 权限也是需要注意的喔！ 我们底下就简单的介绍一下一些基本的安全性管理吧！

- SELinux 的相关议题：

其实就如同第七章 (7.4.5) 里面提到的，我们透过登录档的内容就能够知道如何解决 SELinux 对各个服务所造成的问题了。不过，既然我们知道服务是 Samba 了，能不能找出与 Samba 有关的 SELinux 规则呢？当然可以！基本的 Samba 规则主要有：

```
[root@www ~]# getsebool -a &#124; grep samba
samba_domain_controller --&gt; off   &lt;==PDC 时可能会用到
samba_enable_home_dirs --&gt; off    &lt;==开放用户使用家目录
samba_export_all_ro --&gt; off       &lt;==允许只读文件系统的功能
samba_export_all_rw --&gt; off       &lt;==允许读写文件系统的功能
samba_share_fusefs --&gt; off
samba_share_nfs --&gt; off
use_samba_home_dirs --&gt; off       &lt;==类似用户家目录的开放！
virt_use_samba --&gt; off
```

看吧！几乎所有的规则默认都是关闭的！所以我们需要慢慢的打开啊！目前我们仅会用到用户的家目录以及分享成为可擦写， 不过似乎仅要 samba_enable_home_dirs 那个项目设定妥当即可喔！因此我们可以这样做：

```
[root@www ~]# setsebool -P samba_enable_home_dirs=1
[root@www ~]# getsebool -a &#124; grep samba_enable_home
samba_enable_home_dirs --&gt; on
```

这样用户挂载他们的家目录时 (例如 smb1 使用 //127.0.0.1/smb1/) 就不会出现无法挂载的怪问题了！此外， 由于分享成为 Samba 的目录还需要有 samba_share_t 的类型。那我们还有分享 /home/project 还记得吗？那个目录也需要修订喔！ 这样做看看：

```
[root@www ~]# ll -Zd /home/project
drwxrws---. root users unconfined_u:object_r:home_root_t:s0 /home/project

[root@www ~]# chcon -t samba_share_t /home/project
[root@www ~]# ll -Zd /home/project
drwxrws---. root users unconfined_u:object_r:samba_share_t:s0 /home/project
```

如果你分享的目录不只是 Samba ，还包括 FTP 或者是其他的服务时，那可能就得要使用 public_content_t 这个大家都能够读取的类型才行！若你还有发现任何 SELinux 的问题，请依照 /var/log/messages 里面的信息去修订吧！

- 防火墙议题：利用 iptables 来管理

最简单的管理登入 SAMBA 的方法就是透过 iptables 啦！详细的说明我们已经在第九章防火墙中提过了，所以这里不再详加说明。 要知道的是，如果你仅要针对底下的范围开放 Samba 时，可以这样想：

- 仅针对 192.168.100.0/24, 192.168.1.0/24 这两个网域开放 SAMBA 使用权
- SAMBA 启用的 port UDP: 137, 138 及 TCP: 139, 445；

所以 iptables.allow 规则当中应该要加入这几项：

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.allow
# 加入底下这几行！
iptables -A INPUT -i $EXTIF -p tcp -s 192.168.100.0/24 -m multiport \
         --dport 139,445 -j ACCEPT
iptables -A INPUT -i $EXTIF -p tcp -s 192.168.1.0/24 -m multiport \
         --dport 139,445 -j ACCEPT
iptables -A INPUT -i $EXTIF -p udp -s 192.168.100.0/24 -m multiport \
         --dport 137,138 -j ACCEPT
iptables -A INPUT -i $EXTIF -p udp -s 192.168.1.0/24 -m multiport \
         --dport 137,138 -j ACCEPT
[root@www ~]# /usr/local/virus/iptables/iptables.rule
```

这是很简单很简单的防火墙规则，你必须要依据你的环境自行修改 (通常修改那个 192.168.1.0/24 网段即可！)。 由于 smbd 及 nmbd 并不支持 TCP Wrappers ，所以你也只能透过 iptables 来控制了～

- 防火墙议题： 透过内建的 Samba 设定 (smb.conf)

事实上 Samba 已经有许多防火墙机制啦！那就是在 smb.conf 内的 hosts allow 及 hosts deny 这两个参数。 通常我们只要使用 hosts allow 即可，那么没有写入这个设定项目的其他来源就会被拒绝联机的！这是比较严格的设定。 举例来说，如果你只想要让本机、192.168.100.254, 192.168.100.10, 192.168.1.0/24 使用 SAMBA 而已，那么可以这样写：

```
[root@www ~]# vim /etc/samba/smb.conf
[global]
        # 跟防火墙的议题有关的设定
        hosts allow = 127\. 192.168.100.254 192.168.100.10 192.168.1.
[homes]
....保留原始设定....
[root@www ~]# testparm
[root@www ~]# /etc/init.d/smb restart
```

这个设定值的内容支持部分比对，因此 192.168.1.0/24 只要写出前面三个 IP 段即可 (192.168.1.)。 如此一来不但只有数部主机可以登入我们的 SAMBA 服务器，而且设定值又简单！不像 iptables 写的落落长～ 鸟哥建议在防火墙议题方面，只要使用 iptables 或 hosts allow 其中一项即可，当中又以 hosts allow 较为建议唷！ 当然啦，如果你是针对区网开放的，那么设定 iptables 防火墙反而是比较好的呦！因为不需要更动到 smb.conf 配置文件嘛！让服务的设定变的比较单纯些～

- 文件系统议题：利用 Quota 限制用户磁盘使用

既然网芳是要分享文件系统给用户的，那么想当然尔，各个 Samba 用户们确实会将数据放置到你的 Samba 服务器上嘛！ 那万一单个用户随便上传个数百 GB 的容量到你的 Samba 服务器，而且常常给你随意存取一番， 会不会造成文件系统分配不公或者是带宽方面的问题呢？想想就觉得是『会嘛！』那怎办？就透过 Quota 磁盘配额啊！ 磁盘配额我们在基础篇第三版第十五章已经谈过，在本书第一章 (1.2.2-3) 里面也已经有实作过， 在底下请你依据第一章的后续动作来处理吧！

例题：我们预计分配 smb1, smb2, smb3 在他们自己的家目录下，各拥有 300MB/400MB (soft/hard) 的磁盘配额限量，那该如何做？答：请先依据第一章的 Quota (1.2.2-3) 相关数据处理完：

- /etc/fstab 加入 /home 挂载点的 usrquota,grpquota 等设定值；
- 重新挂载 /home ，让 Quota 实际被支持；
- 以 quotacheck -avug 建立 Quota 的数据库档案；
- 启动 Quota ；

假若你已经于第一章就处理完毕了，那么这一题就非常简单喔！透过 edquota -u smb1 来处理即可！

```
[root@www ~]# edquota -u smb1
Disk quotas for user smb1 (uid 2004):
  Filesystem                   blocks    soft    hard inodes   soft   hard
  /dev/mapper/server-myhome         0 300000  400000       0      0      0

[root@www ~]# edquota -p smb1 smb2
[root@www ~]# edquota -p smb1 smb3
[root@www ~]# repquota -ua
*** Report for user quotas on device /dev/mapper/server-myhome
Block grace time: 7days; Inode grace time: 7days
                        Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
smb1      --      32  300000  400000              9     0     0
smb2      --      32  300000  400000              8     0     0
smb3      --      32  300000  400000              8     0     0
```

16.2.7 主机安装时的规划与中文扇区挂载

现在你知道 Samba 服务器的功能是用来作为文件服务器的，每个使用者都可以拥有家目录，并透过网芳的功能来链接到 Samba 服务器中。这就有个问题啦，那就是你的使用者如果太多，并且将他们的重要数据都放到这部 Samba 服务器上头的话，那肯定 /home 未来会有点不足啊！所以 /home 所在的磁盘或许可以使用大一点的硬盘， 或者使用磁盘阵列，使用 LVM (基础学习篇第三版十五章) 也是个不错的方案。 底下为简单的思考方向：

- 在安装 Linux 的时候，建议不需要安装 X Window ；
- 在规划 Linux 时，/home 最好独立出一个 partition ，而且硬盘空间最好能够大一些；
- /home 独立出来的 partition 可以单独进行 quota 的作业，以规范用户的最大硬盘用量；
- 无网卡的打印机 (USB) 可直接链接到 Linux 主机再透过 Samba 分享；
- 由于 SAMBA 一般来说都仅针对内部 (LAN) 主机进行开放，所以，可能的话 SAMBA 主机直接使用 private IP 来设定即可，当然啦， SAMBA 是否使用 private IP 还得视你的整个网域的 IP 网段的特性来规划。 以鸟哥研究室来说，因为实验室所有计算机的 IP 都是 Public IP，那么 SAMBA 如果使用 Private IP 反而会让大家都无法连接上啊！ ^_^
- 如果你的 SAMBA 主机使用 Public IP 时，请特别留意规范好防火墙的设定，尽量仅让 LAN 内的计算机可以联机进来即可，不要对 Internet 开放喔！

另外，如果你的 Samba 服务器需要挂载含有中文的 partition 时，譬如说你将原本 Windows XP 的 FAT32 文件系统挪到 Linux 系统下，此时如果用一般模式来挂载该分割槽时，一些中文档名可能会无法被顺利的显示出来。 这个时候你就得需要这样做了：

```
mount -t vfat -o iocharset=big5,codepage=950 /dev/sd[a-p][1-15] /mount/point
```

其中 iocharset 指的是本机的语系编码方式， codepage 则与远程软件有关。因为我们是在本机进行挂载， 所以实际上使用 iocharset 这个参数即可啦！更多说明则请看下节的客户端设定部分啰！

16.3 Samba 客户端软件功能

现在你已经架设好了 Samba 服务器啦！有服务器当然要有客户端来使用才是好的服务器嘛！不然要这个服务器干嘛？ 而我们假设局域网络内有 Windows/Linux 系统，这两种系统都是透过 NetBIOS over TCP/IP 来连上 Samba 服务器的， 在设定之前你必须要知道的有几件事：

- 在区网内的主机最好具有相同的工作组，且具有不同的主机名；
- Windows XP pro. 最多仅能允许十个用户同时连接到自己的网芳；
- 你可以在网芳当中看到的通常是相同群组的主机；
- 可以使用『搜寻』-->『计算机』-->『输入 IP』来查到 Samba 主机；
- Windows 的网芳预设仅有同一 IP 网段的主机才能登入 (Windows 防火墙设定)！

接下来咱们就分别依照 Windows 及 Linux 系统来做说明吧！

16.3.1 Windows 系统的使用

在 Windows 上面的搜寻网络上的网芳主机实在挺简单的，你有好几种方法可以处理：

- 打开『档案总管』，『网络上的芳邻』、『整个网络』、『Microsoft Windows Network』 就能看到属于你群组的所有计算机主机了！
- 『开始』、『搜寻』、『档案或文件夹』、『计算机或人员』、『网络上的计算机』，然后在出现的方框当中填写正确的 IP ，按下『搜寻』即可！这个方法可以适用于不在同一个群组当中的网络主机喔！
- 如果是 Windows 7 的话，只要点选文件夹即可。

举例来说，如果想要连接到我们的 Samba 主机的话，而又不知到这部 Samba 主机的 NetBIOS name ， 那利用搜寻的结果会有点类似如下的图示：

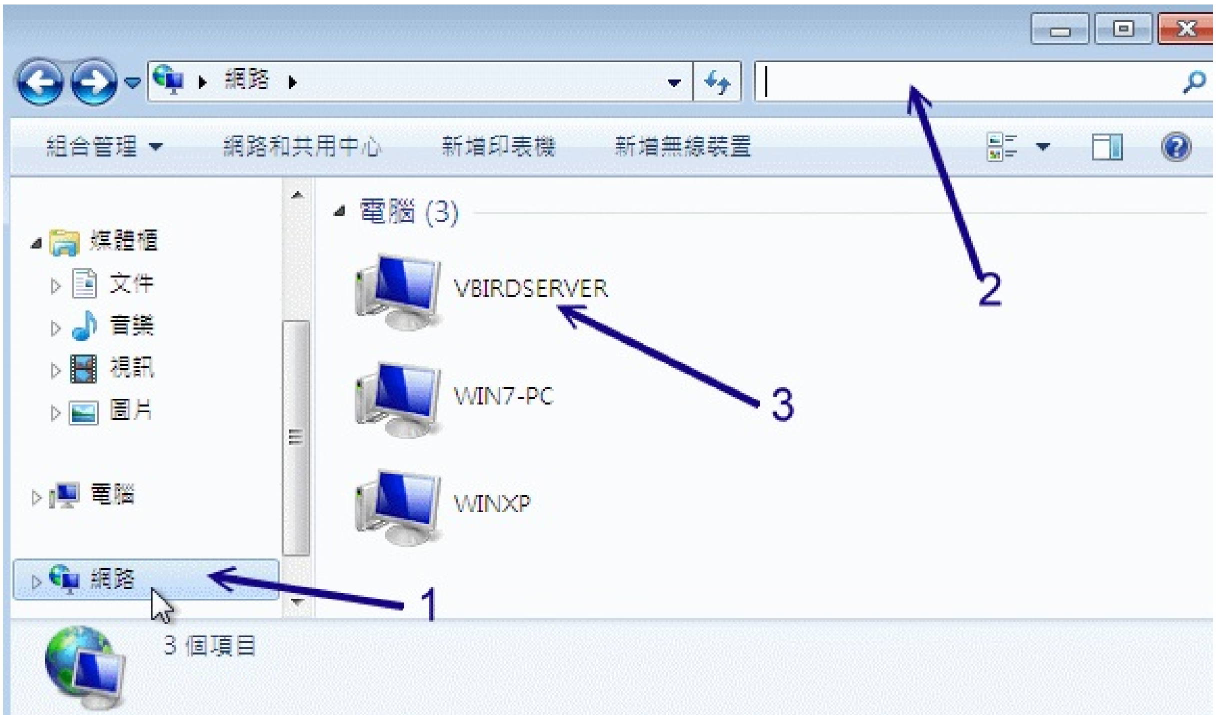


图 16.3-1、Windows 7 客户端搜寻示意图

上图左侧先点选『网络』，然后到右上方的框框中，输入 NetBIOS name，若不知道的话，就留白让 Windows 7 自己找。 如上图所示，就有找到三部网络主机啊！我们来点选一下 VBIRDSERVER 吧！ 因为要登入人家服务器，所以就被要求要输入密码。 如下图所示，请填写好你所拥有的账号与密码吧！

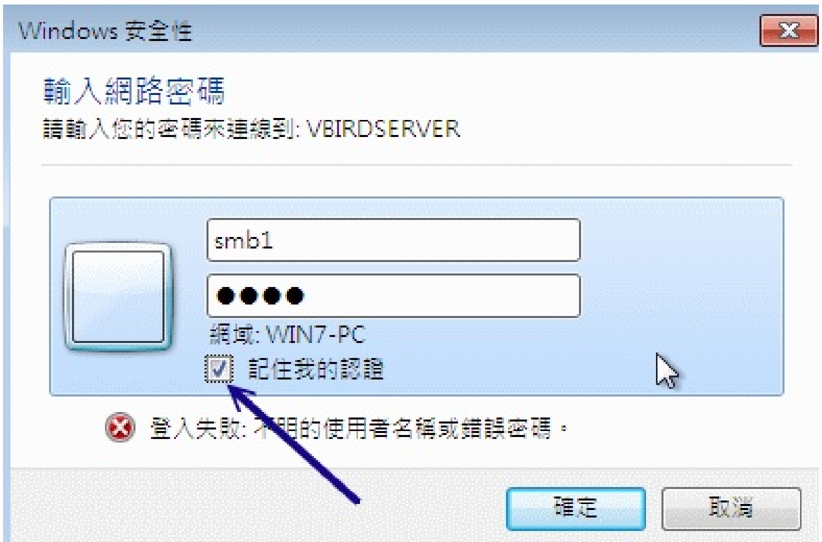


图 16.3-2、Windows 7 客户端登入 SAMBA 服务器示意图

若顺利登入系统了，那么就能够看到如下的图示，就是取得该服务器的可用资源啦！因为我们并没有针对 Windows 7 提供打印机的驱动程序，那部份先略过。我们现在来将 project 挂载成本机磁盘试看看：

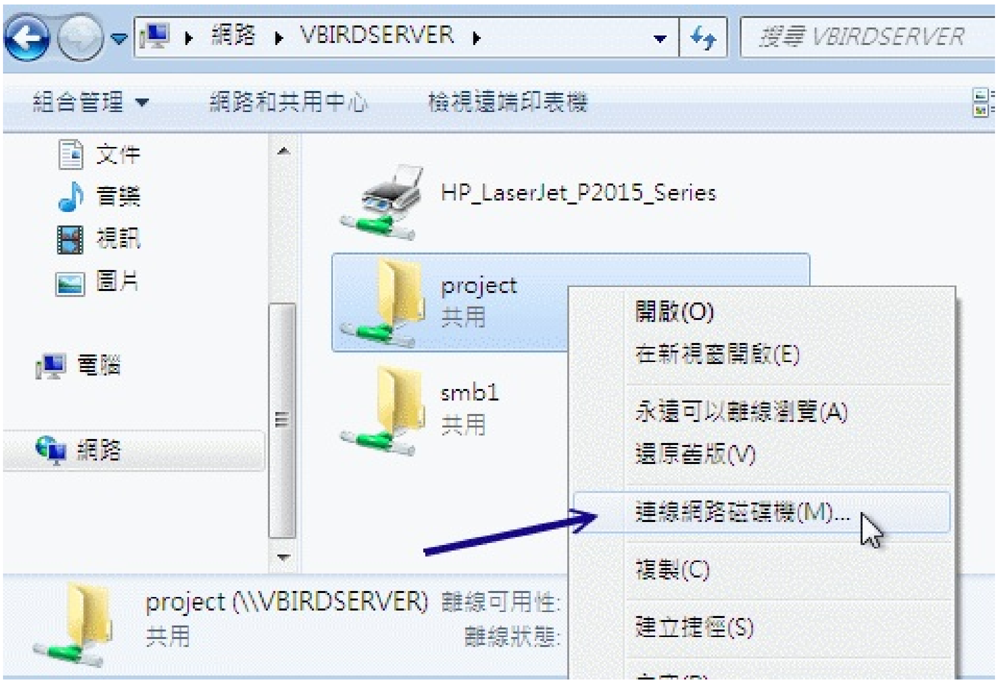


图 16.3-3、Windows 7 客户端登入 SAMBA 服务器示意图

如上图所示，在 project 上面右键单击，选择『联机网络驱动器机』，就会出现如下的画面让你去选择挂载磁盘驱动器的参数喔：

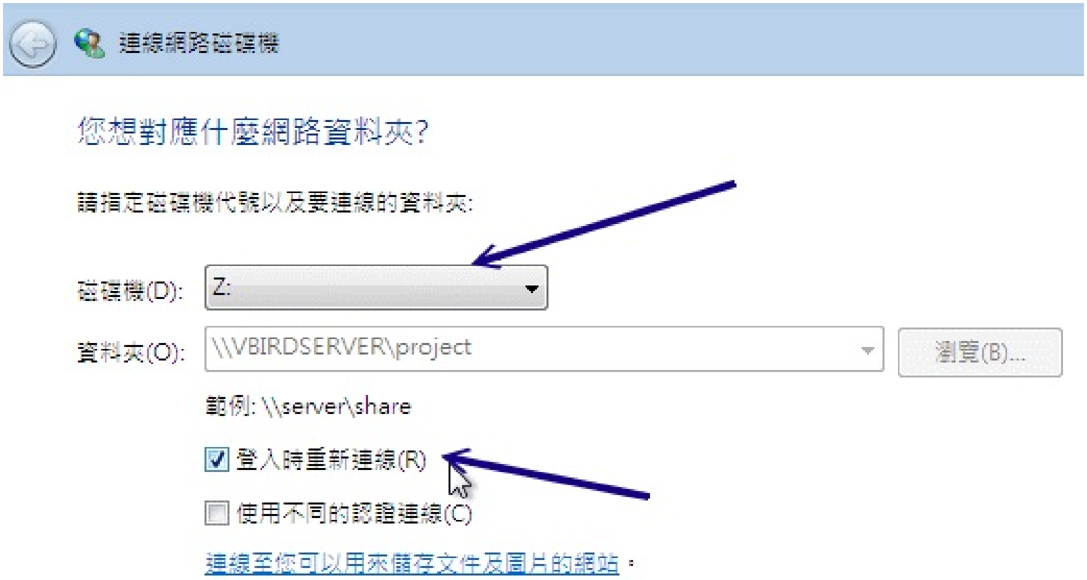


图 16.3-4、Windows 7 客户端挂载网络驱动器机的示意图

你可以自己调整想要的驱动器号，例如预设的 Z 槽， 那么以后你的档案总管中就会生出一个 Z 槽，该磁盘槽就代表 \192.168.100.254\project 那个分享的目录啰！

- 让 Windows 系统的网芳支持不同网域的 IP 联机

由于网芳的资安问题越来越严重，因此 Windows XP 之后的版本都预设仅开放本机 IP 网域的网芳联机而已。 如果你的 Windows 想要让别人可以在 Internet 或不同的 IP 网段对你联机时，你就得修改一下防火墙的设定啊！ 请叫出控制台，然后点选『Windows 防火墙』就会出现如下的图示了：

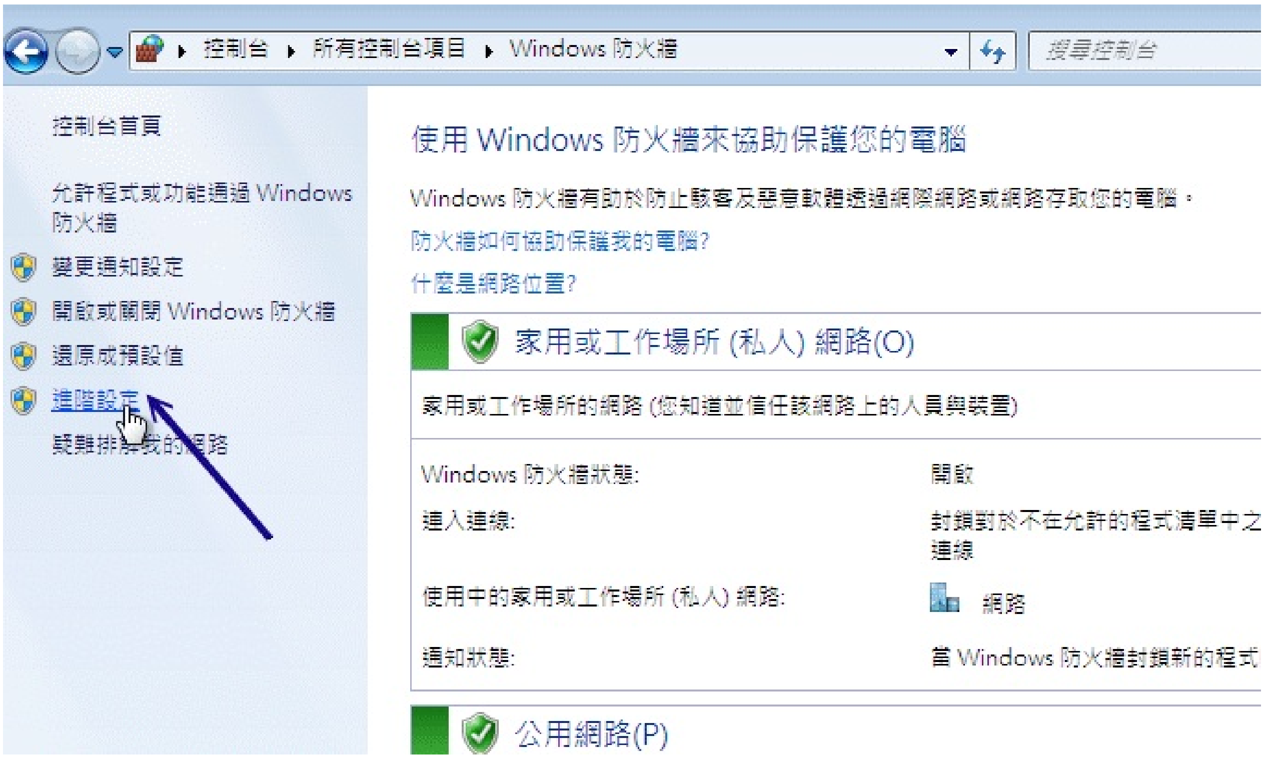


图 16.3-5、Windows 7 服务器防火墙示意图

因为我们得要细部设定防火墙，因此点选上图中左侧的『进阶设定』来取得如下图示吧！

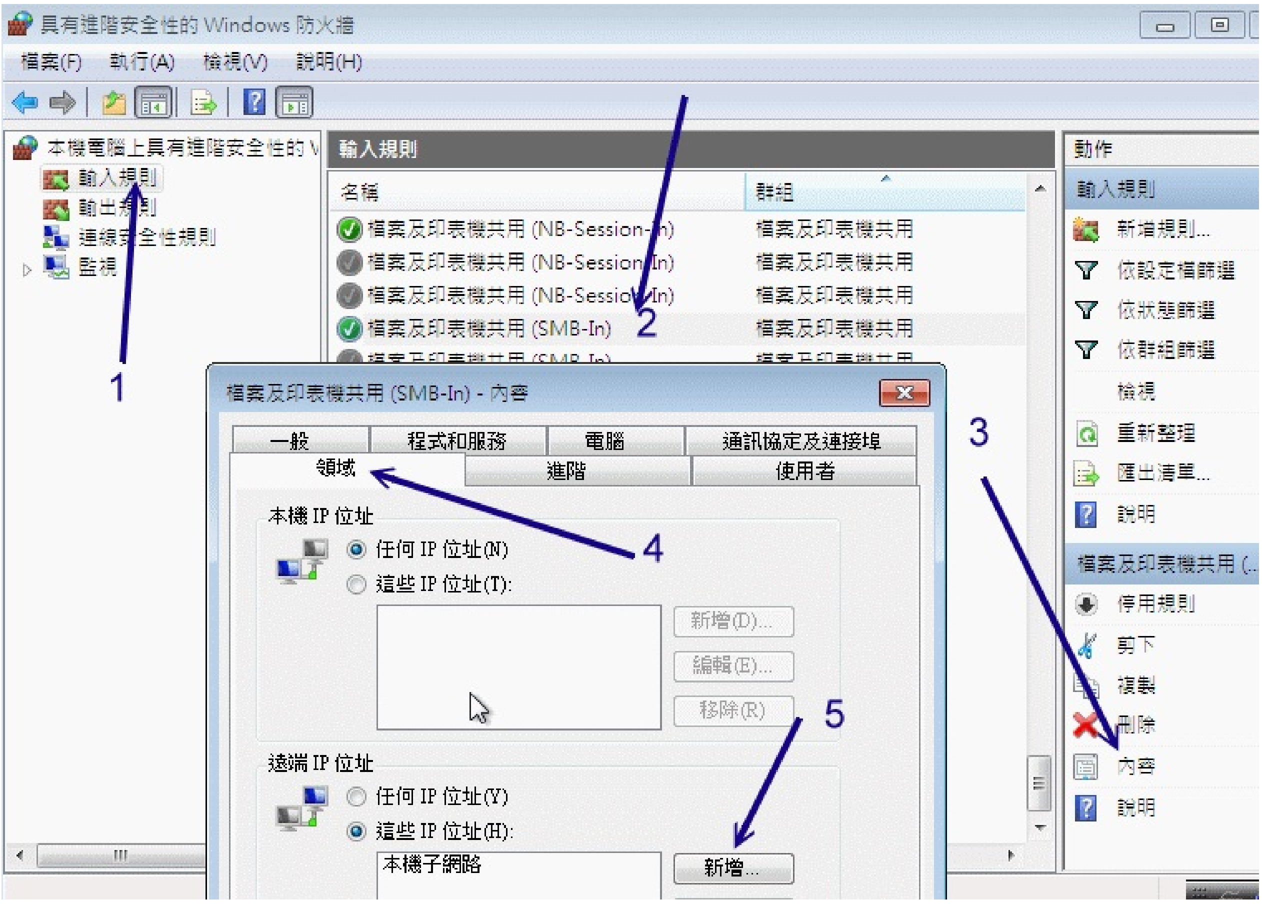


图 16.3-6、Windows 7 服务器防火墙示意图

还记得网络是双向的吧？所以，我们得先要针对输入 (从外部连到本机) 的规则来处置。如上图所示，按下 (1)输入规则，然后点选 (2)档案及打印机共享，之后到 (3)选择详细的规则内容，会出现另外一个窗口，在 (4)点选『领域』的部分来设定不同网段， 最终在 (5)的地方『新增』可进入本机的远程 IP 网段喔！按下新增会出现如下图示喔：

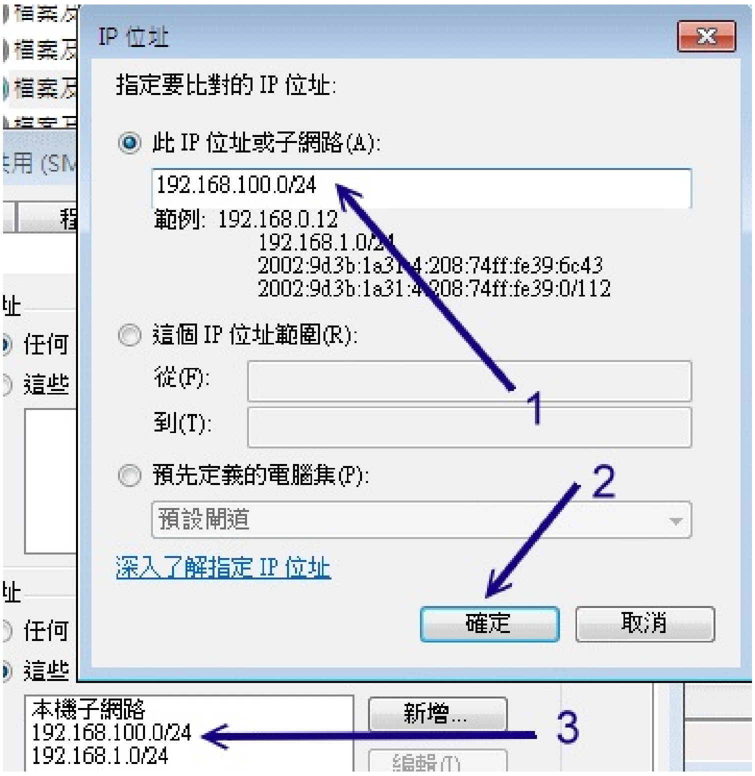


图 16.3-7、Windows 7 服务器防火墙示意图

如上图所示，在 (1)填写正确的 IP 或网段，然后按下 (2)确定后，就能够在 (3)的框框当中出现可联机的远程服务器啰！

- 透过 port 445 的特殊登入方式

如果你知道 Samba 服务器有启用 port 445 ，并且他已经分享了某个目录时，举例来说，我们的 192.168.100.254 有分享出 project 这个分享资源名称时，那么这个目录的完整写法为：『 \\192.168.100.254\project 』， 我们可以透过『开始』出现的那个方框来处理这个玩意儿！如下所示：

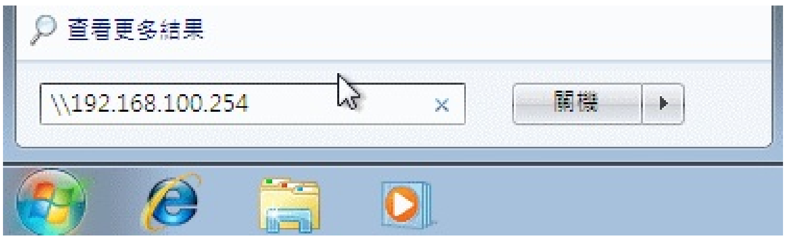


图 16.3-8、Windows 7 透过 port 445 联机

如果可以登入的话就会顺利登入，否则就会弹出一个要你输入账号密码的窗口，输入正确的数据即可！ 呼呼！真过瘾～除此之外，我们还可以登入别人 Windows 主机的 C 或 D 槽喔！写法则变成这样：

- \192.168.100.20\c$

所以说，怕了吧！俺还真害怕～所以啊，Samba 没必要时，那个 port 445 应该是可以关闭的吧！

16.3.2 Linux 系统的使用

- smbclient：查询网芳分享的资源，以及使用类似 FTP 的方式上传/下载网芳

咱们的 Samba 有提供 Linux 网芳的客户端功能喔！也就是说 Linux 可以挂载 Samba 服务器也能挂载 Windows 提供的网芳啦！主要是透过 smbclient 来观察，再以 mount 来挂载文件系统哩。先来介绍一下 smbclient 这个指令吧：

```
# 1\. 关于查询的功能，例如查出 192.168.100.254 的网芳数据
[root@clientlinux ~]# smbclient -L //[IP&#124;hostname] [-U username]
[root@clientlinux ~]# smbclient -L //192.168.100.254 -U smb1
Enter smb1's password:
Domain=[VBIRDHOUSE] OS=[Unix] Server=[Samba 3.5.4-68.el6_0.2]

        Sharename       Type      Comment
        ---------       ----      -------
        project         Disk      smbuser's project
        print$          Disk      Printer drivers
        IPC$            IPC       IPC Service (This is vbird's samba server)
        HP_LaserJet_P2015_Series Printer   HP LaserJet P2015 Series
        smb1            Disk      Home Directories &lt;==等一下用这个当范例
Domain=[VBIRDHOUSE] OS=[Unix] Server=[Samba 3.5.4-68.el6_0.2]

        Server               Comment
        ---------            -------
        VBIRDSERVER          This is vbird's samba server

        Workgroup            Master
        ---------            -------
        VBIRDHOUSE           VBIRDSERVER
# 从这里可以知道在目前网域当中有多少个工作组与主要的名称解析主机
```

除了这个先前用过的查询功能之外，我们可以这样简易使用网芳的：

```
# 2\. 利用类似 FTP 的方式登入远程主机
[root@clientlinux ~]# smbclient '//[IP&#124;hostname]/资源名称' [-U username]
# 意思是使用某个账号来直接登入某部主机的某个分享资源，举例如下：
[root@clientlinux ~]# smbclient '//192.168.100.254/smb1' -U smb1
Enter smb1's password:
Domain=[VBIRDHOUSE] OS=[Unix] Server=[Samba 3.5.4-68.el6_0.2]
smb: \&gt; dir
# 在 smb: \&gt; 底下其实就是在 //192.168.100.254/dmtsai 这个目录底下啦！所以，
# 我们可以使用 dir, get, put 等常用的 ftp 指令来进行数据传输了！
?   :列出所有可以用的指令，常用！
cd  :变换到远程主机的目录
del :杀掉某个档案
lcd :变换本机端的目录
ls  :察看目前所在目录的档案
dir :与 ls 相同
get :下载单一档案
mget:下载大量档案
mput:上传大量档案
put :上传单一档案
rm  :删除档案
exit:离开 smbclient 的软件功能
# 其他的指令用法请参考 man smbclient 喔！
```

- mount.cifs：直接挂载网芳成为网络驱动器机

事实上，使用 smbclient 一点也不方便，因为使用的是 ftp 的功能语法，有点怪怪的～能不能像 Windows 那样，可以直接联机网络驱动器机啊？这当然没有问题！不过就需要藉由 mount.cifs 来协助了！

早期的 Samba 主要是提供 smbmount 或 mount.smbfs 这个指令来挂载 (smbfs 是 SMB filesystem 的缩写)， 不过这个指令已经被可以进行比较好的编码判断的 mount.cifs 所取代啦！mount.cifs 可以将远程服务器分享出来的目录整个给他挂载到本机的挂载点，如此一来，远程服务器的目录就好像在我们本机的一个分割槽一样喔！可以直接执行复制、编辑等动作！这可就好用的多了！ 底下我们来谈一谈怎么用这个 mount.cifs 吧！

```
[root@clientlinux ~]# mount -t cifs //IP/分享资源 /挂载点 [-o options]
选项与参数：
-o 后面接的参数 (options) 常用的有底下这些：
   username=你的登入账号：例如 username=smb1
   password=你的登入密码：需要与上面 username 相对应啊！
   iocharset=本机的语系编码方式，如 big5 或 utf8 等等；
   codepage=远程主机的语系编码方式，例如繁体中文为cp950

# 范例一：以 smb1 的身份将其家目录挂载至 /mnt/samba 中
[root@clientlinux ~]# mkdir /mnt/samba
[root@clientlinux ~]# mount -t cifs //192.168.100.254/smb1 /mnt/samba \
&gt; -o username=smb1,password=4321,codepage=cp950
[root@clientlinux ~]# df
文件系统              1K-区段      已用      可用 已用% 挂载点
//192.168.100.254/smb1/
                       7104632    143368   6606784   3% /mnt/samba
```

经由 mount 的动作，我们就可以轻易的将远程分享出来的咚咚给他挂载到自己 Linux 本机上面！好用的很～更详细的 mount 用法，请 man mount！

- nmblookup：查询 NetBIOS name 与 IP 及其他相关信息：

现在我们可以透过一些 NetBIOS 相关的功能来取得 NetBIOS name ，不过，如果你还想要知道这个 NetBIOS name 的其他信息时， 例如 IP、分享的资源等等，那可以使用 nmblookup 这个指令来搞定即可。他是这么使用的：

```
[root@clientlinux ~]# nmblookup [-S] [-U wins IP] [-A IP] name
选项与参数：
-S  ：除了查询 name 的 IP 之外，亦会找出该主机的分享资源与 MAC 等；
-U  ：后面一般可接 Windows 的主要名称管理服务器的 IP ，可与 -R 互用；
-R  ：与 -U 互用，以 Wins 服务器来查询某个 Netbios name；
-A  ：相对于其他的参数， -A 后面可接 IP ，藉 IP 来找出相对的 NetBIOS 数据；

# 范例一：藉由 192.168.100.254 找出 vbirdserver 这部主机的 IP 地址
[root@clientlinux ~]# nmblookup -U 192.168.100.254 vbirdserver
querying vbirdserver on 192.168.100.254
192.168.100.254 vbirdserver&lt;00&gt;
192.168.1.100 vbirdserver&lt;00&gt;    &lt;==之前鸟哥就说有两个 IP 嘛！俺的主机！

# 范例二：找出 vbirdserver 的 MAC 与 IP 等信息：
[root@clientlinux ~]# nmblookup -S vbirdserver
querying vbirdserver on 192.168.100.255  &lt;==在区网内广播开始找！
192.168.100.254 vbirdserver&lt;00&gt;          &lt;==找到 IP 啰！
Looking up status of 192.168.100.254
        VBIRDSERVER     &lt;00&gt; -         B &lt;ACTIVE&gt;
        ..__MSBROWSE__. &lt;01&gt; - &lt;GROUP&gt; B &lt;ACTIVE&gt;
        VBIRDHOUSE      &lt;00&gt; - &lt;GROUP&gt; B &lt;ACTIVE&gt;
```

- smbtree：网络上的芳邻浏览器显示模式！

如果你想要使用类似 Windows 上面，可以一看就明了各个网芳所分享的资源时，你能使用 smbtree 来直接查询喔！ 这个指令更简单！直接输入就能用：

```
[root@clientlinux ~]# smbtree [-bDS]
选项与参数：
-b  ：以广播的方式取代主要浏览器的查询
-D  ：仅列出工作组，不包括分享的资源
-S  ：列出工作组与该工作组下的计算机名称 (NetBIOS) 不包括各项资源目录

# 范例一：列出目前的网芳树状相关图
[root@clientlinux ~]# smbtree
Enter root's password:  &lt;==直接按 [Enter] 即可！
WORKGROUP
        \\WIN7-PC
VBIRDHOUSE
        \\WINXP
cli_start_connection: failed to connect to WINXP&lt;20&gt; (0.0.0.0).
        \\VBIRDSERVER                   This is vbird's samba server
             \\VBIRDSERVER\HP_LaserJet_P2015_Series  HP LaserJet P2015 Series
             \\VBIRDSERVER\IPC$     IPC Service (This is vbird's samba server)
             \\VBIRDSERVER\print$   Printer drivers
             \\VBIRDSERVER\project  smbuser's project

[root@clientlinux ~]# smbtree -S
Enter root's password:
WORKGROUP
        \\WIN7-PC
VBIRDHOUSE
        \\WINXP
        \\VBIRDSERVER                   This is vbird's samba server
# 此时仅有工作组与计算机名称而已呢！
```

- smbstatus：观察 SAMBA 的状态

其实这个指令算是服务器的相关功能啦！因为它主要的目的是查阅目前 SAMBA 有多少人来联机， 且哪些资源共享已经被使用等等的信息。所以如果你想要使用这个软件，请先安装 samba 喔！简单用法如下：

```
[root@www ~]# smbstatus [-pS] [-u username]
选项与参数：
-p ：列出已经使用 SAMBA 联机的程序 PID ；
-S ：列出已经被使用的资源共享状态；
-u ：只列出某个用户相关的分享数据

# 范例一：列出目前主机完整的 Samba 状态
[root@www ~]# smbstatus
Samba version 3.5.4-68.el6_0.2
PID     Username      Group         Machine
-------------------------------------------------------------------
5993      smb1          smb1          __ffff_192.168.100.10 (::ffff:192.168.100.10)
5930      smb1          smb1          win7-pc      (::ffff:192.168.100.30)
# 上半部主要在列出目前联机的状态中，主要来自那个客户端机器与登入的用户名

Service      pid     machine       Connected at
-------------------------------------------------------
IPC$         5930   win7-pc       Fri Jul 29 15:56:03 2011
project      5930   win7-pc       Fri Jul 29 15:59:25 2011
smb1         5993   __ffff_192.168.100.10  Fri Jul 29 16:32:45 2011
# 这部分则显示出，目前有几个目录被使用了？那个 smb1 代表 //IP/smb1/ 喔！
```

你可以透过这个小程序来了解到目前有多少人使用你的 SAMBA 的啦！

16.4 以 PDC 服务器提供账号管理

我们在 16.1.5 约略谈过 PDC 这个玩意儿，他可以让用户在计算机教室的任何一个地方，都用同一组账号密码登入， 并可取得相同的家目录等数据，这与我们之前谈到的，在 Linux 底下使用 NIS 搭配 NFS 是很类似的作法！ 只是它是用在 Windows 上头就是了。那如何完成呢？我们底下就来谈谈这个玩意儿！^_^

16.4.1 让 Samba 管理网域使用者的一个实作案例

前面介绍的内容都是属于 Peer/Peer 的联机状况，也就是 Samba 服务器与 Windows 客户端其实是平等地位的啦！ 所以 Windows 客户端需要知道 Samba 服务器内的账号密码数据后，才能够顺利的使用 Samba 的资源。 不过，这样的方式在较大型一些的局域网络环境可能就会有点困扰，例如学校的环境。

举例来说，如果你有一个计算机教室里面有 50 部 Windows XP Pro. 的个人计算机，由于计算机教室大家都会使用， 因此里面这 50 部个人计算机有使用还原精灵，也就是每次计算机重新启动后整个操作系统就会还原成原本的样子。 但我们知道使用者总是需要有个人家目录吧？他们总不希望这次的工作在重新启动后就失去了～ 所以我们可以利用一部主机来让他们储存数据啊！那就是 Primary Domain Controller (PDC) 服务器。

其实 Samba PDC 的作用很简单，就是让 Samba PDC 成为整个局域网络的领域管理员 (domain controller)， 然后让 Windows 主机加入这个领域，未来使用者利用 Windows 登入时，(1)Windows 会前往 PDC 服务器取得用户的账号密码， 同时 (2)PDC 还会传送用户的重要数据到那部 Windows 个人计算机上，而 Windows 计算机上的用户注销时， (3)该用户修改过的数据也会回传给 PDC 。如此一来不管这个使用者在哪一部个人计算机上面登入， 他都能够取得正确的个人资料！很棒的作用吧！

PDC 是个很复杂的环境，他可以达到的功能相当的多，而且密码的验证也不必在同一部 PDC 主机上面， 不过这里我们不谈那么复杂的东西，只是做一个简单的练习，因此底下的这部 PDC 使用 Linux 自己的密码来进行验证， 并且也只管理自己所分享出去的资源啰！至于假设网络环境与相关工作组参数如下：

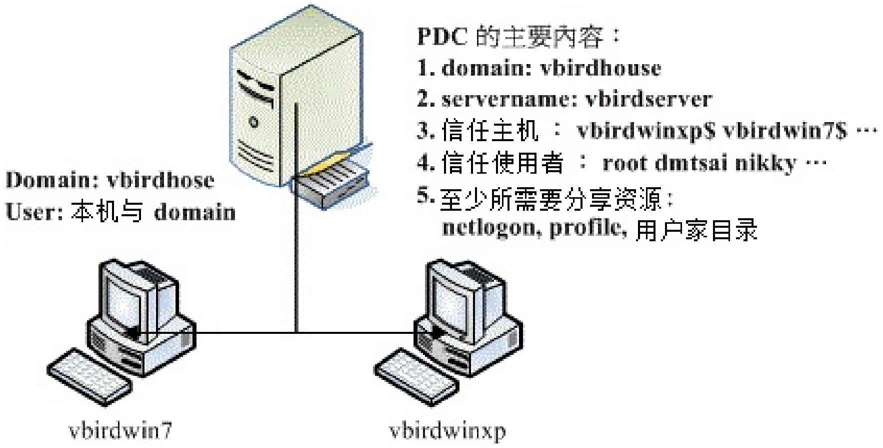


图 16.4-1、一个简易的 PDC 实作案例相关参数示意图

整个基本的设定流程应该是这样的：

- 区网计算机环境设定：整体网域设定好，尤其 Windows 的工作组与计算机名称及 IP 等参数；
- PDC 设定：因为 PDC 管理自己的密码，所以 security = user；
- PDC 最好拥有整个网域的名称解析权力，亦即成为主要的名称解析器；
- 需有 netlogon 资源共享，提供 windows 2000/XP pro. 客户端的登入之用；
- 由于 Windows 需读入个人配置文件，默认目录为 profile，Linux 系统需预先设定此目录；
- 增加 PDC 上的使用者账号以及机器代码 (machine account) 等等
- 在 Windows 2000/XP pro. 个人计算机上设定成为 PDC 的客户端。

底下咱们就来依序处理处理先！

16.4.2 PDC 服务器的建置

PDC 服务器的建立非常的麻烦，需要一步一步的实作进行，挺讨厌的。而且，由于建置 PDC 的环境主要在管理整个区网内的 Windows 计算机，因此每部 Windows 计算机的主机名与相关参数要先确定下来，如同上一小节的图示内， 每部计算机的角色定位都需要清楚才行。清楚了各个计算机的角色后，接下来就能够慢慢的实作进行啰！

- 1. 建置 NetBIOS 与 IP 对应的数据：设定 lmhosts 与 /etc/hosts

由于我们的 Samba 即将成为整个网域的名称解析者，因此你最好将整个网域的 NetBIOS name 与 IP 的对应写入 lmhosts 档案当中。如果你的区网是以 DHCP 发放 IP 的，那么你最好搭配 DNS 系统去建置你的主机名对应信息， 否则主机名对应不起来，总是有点困扰。在这个案例中，由于鸟哥使用的 NetBIOS name (如 vbirdserver) 与主机名 (如 www.centos.vbird) 并不相同，因此这里建议需要修改 lmhosts 才好。

```
[root@www ~]# vim /etc/samba/lmhosts
127.0.0.1       localhost      &lt;==这行是预设存在的，不要动他，底下的请自行新增
192.168.100.254  vbirdserver
192.168.100.10   vbirdlinux
192.168.100.20   vbirdwinxp
192.168.100.30   vbirdwin7

[root@www ~]# vim /etc/hosts
192.168.100.254 www.centos.vbird        vbirdserver
192.168.100.10  clientlinux.centos.vbird     vbirdlinux
192.168.100.20  vbirdwinxp
192.168.100.30  vbirdwin7
```

由于 Linux 上的 Samba 很多数据还是与 TCP/IP 的主机名有关，所以除了 lmhosts 之外，建议还是处理一下 /etc/hosts 比较妥当！这样就行啦！

- 2. 建置 PDC 主设定：处理 smb.conf

假设我们要让 PDC 客户端登入时可以取得他自己的家目录，那么需要这样处理：

```
[root@www ~]# vim /etc/samba/smb.conf
[global]
        workgroup       = vbirdhouse   &lt;==请务必确认一下工作组与主机名
        netbios name    = vbirdserver
        server string   = This is vbird's samba server
        unix charset    = utf8
        display charset = utf8
        dos charset     = cp950
        log file        = /var/log/samba/log.%m
        max log size    = 50
        security        = user
        passdb backend  = tdbsam
        load printers   = yes
        cups options    = raw
        printcap name   = cups
        printing        = cups

        # 与 PDC 有关的一些设定值：
        # 底下几个设定值处理成为本局域网络内的主要名称解析器
        preferred master = yes
        domain master    = yes
        local master     = yes
        wins support     = yes
        # 操作系统 (OS) 等级越高才能成为主网域的控制者，一般 NT 为 32,
        # Windows 2000 为 64 ，所以这里我们设定高一点，但不可超过 255
        os level       = 100
        # 底下则是设定能否利用 PDC 登入，且登入需要进行哪些动作：
        domain logons = yes
        logon drive   = K:              &lt;==登入后家目录挂载成 Windows 哪一槽
        logon script  = startup.bat     &lt;==每个使用者登入后会自动执行的程序
        time server   = yes             &lt;==自动调整 Windows 时间与 Samba 同步
        admin users   = root            &lt;==预设的管理员账号！预设为 root
        logon path    = \\%N\%U\profile &lt;==使用者的个人化设定
        logon home    = \\%N\%U         &lt;==用户的家目录位置！

# 这个在指定登入者能够进行的工作，里面主要是具有许多执行程序：
[netlogon]  &lt;==与前面的 logon script 有关，该程序放置在这里
   comment         = Network Logon Service
   path            = /winhome/netlogon  &lt;==重要的目录，要自己建立才行！
   writable        = no
   write list      = root
   follow symlinks = yes
   guest ok        = yes

[homes]
....(底下保留原本设定)....

[root@www ~]# testparm
[root@www ~]# /etc/init.d/smb restart
[root@www ~]# /etc/init.d/nmb restart
```

上面的设定有几个地方比较有趣一点：

- time server：要使 Samba 与 Windows 主机的时间同步，使用这个项目；
- logon script：当使用者以 Windows 客户端登入后，Samba 可以提供一支批处理文件，让使用者去设定好他们自己的目录配置。整个配置的内容记录在 startup.bat 当中。 你要注意的是，这个 startup.bat 档名可以随意更改，不过他必须要放置到 [netlogon] 所指定的目录内；
- logon drive：那么这个家目录要挂载到那个分割槽？ 在 Windows 底下大多以 C, D, E... 做为磁盘的代号，你这里可以指定一下家目录要放置成为那个磁盘代号；
- admin users：指定这个 Samba PDC 的管理员身份。
- [netlogon]：指定利用网络登录时首先去查询的目录资源。

- logon path：用户登入后，会取得的环境设定数据在哪？ 我们知道用户会有一堆环境数据，例如桌面等，这些东西都放置到这里来。使用的变量中， %N 代表 PDC 服务器的位置， %U 则代表用户的 Linux 家目录。因此最终你得要有 ~someone/profile 的目录才可以。
- logon home：用户的家目录，默认与 Linux 的家目录相同位置。

- 3. 建立 Windows 客户端登入时所需的设定数据 netlogon 目录

先来建立 [netlogon] 内所需要的数据好了，那就是一个目录。由于鸟哥预计将所有的 PDC 数据通通放置到 /winhome 当中，包括用户家目录，因此很多东西都需要修订喔！包括后来的 SELinux 肯定会出问题的～

```
[root@www ~]# mkdir -p /winhome/netlogon
```

接下来我们还得要建立允许使用者执行的档案，就是那个 startup.bat 才行！ 注意一下，我们这里假设用户家目录为 K 槽，那你可以这样做：

```
[root@www ~]# vim /winhome/netlogon/startup.bat
net time \\vbirdserver /set /yes
net use K: /home
# 这个档案的格式为：net use [device:] [directory]

# 再将该档案转成 DOS 的断行格式才行！因为是提供给 Windows  系统嘛！
[root@www ~]# yum install unix2dos
[root@www ~]# unix2dos /winhome/netlogon/startup.bat
[root@www ~]# cat -A /winhome/netlogon/startup.bat
net time \\vbirdserver /set /yes^M$
net use K: /home^M$
# 瞧见吗？会多出个奇怪的 ^M 符号，那就是 Windows 断行字符。
```

- 4. 建立 Windows 专用的使用者

因为鸟哥预计将使用者全部挪到 /winhome 底下，而且每个用户家目录应该还要有 profile 目录存在才行， 为了避免麻烦，所以我们先到 /etc/skel 去处理一下，然后才建立账号，最后才产生 samba 用户吧！ 产生 samba 用户可以使用 pdbedit 也能够直接使用 smbpasswd -a ，因为没有要用特殊的参数， 所以，Samba 用户就用旧的 smbpasswd 来处理即可。

```
[root@www ~]# mkdir /etc/skel/profile
[root@www ~]# useradd -d /winhome/dmtsai dmtsai
[root@www ~]# useradd -d /winhome/nikky  nikky
[root@www ~]# smbpasswd -a root
[root@www ~]# smbpasswd -a dmtsai
[root@www ~]# smbpasswd -a nikky
[root@www ~]# pdbedit -L
smb1:2004:
smb3:2006:
smb2:2005:
student:505:
root:0:root
dmtsai:2007:
nikky:2008:
# 重点是需要有画底线的那几个人物出现才行呦！

[root@www ~]# ll /winhome
drwx------. 5 dmtsai dmtsai 4096 Jul 29 16:49 dmtsai
drwxr-xr-x. 2 root   root   4096 Jul 29 16:48 netlogon
drwx------. 5 nikky  nikky  4096 Jul 29 16:49 nikky
# 用户的家目录不是在 /home 而是在 /winhome 里头才是对的呦！
```

那以后新增的使用者都有可以存放来自 Windows 的特殊配置文件目录喔！比较好管理啰～ 当然啦，使用 useradd 新增使用者后，记得也要使用 smbpasswd -a username 来让该使用者可以使用 Samba 喔！

- 5. 建立机器码账号

由于 PDC 会针对 Windows 客户端的主机名 (NetBIOS name) 进行主机账号检查， 所以我们也要为客户端的主机名进行账号的设定。咦！啥是主机账号？ 一般用户账号是英文或数字，主机账号则在该账号最后面加上一个钱字号『$』即可！ 举例来说， vbirdwinxp 这部主机可设定的账号名称为 vbirdwinxp$。

而我们知道要使用 smbpasswd 增加的使用者必须要在 /etc/passwd 当中，因此要建立这个账号你就得要这样做：

```
[root@www ~]# useradd -M -s /sbin/nologin -d /dev/null vbirdwinxp$
[root@www ~]# useradd -M -s /sbin/nologin -d /dev/null vbirdwin7$
```

会增加 -M -s -d 等参数的原因是因为不想要让这个账号具有可以登入的权限， 因此将这个主机账号设定的比较怪一点～ ^_^～接下来让 Samba 知道这个账号是主机账号，所以你应该要这样做：

```
[root@www ~]# smbpasswd -a -m vbirdwinxp$
[root@www ~]# smbpasswd -a -m vbirdwin7$
```

这样便加入主机账号啰！而我们的 Samba PDC 也就可以透过『主机账号』来判断 Windows 客户端能否连上来， 若连接上 PDC 与 Windows 客户端后，接下来一般使用者账号就可以在 windows 客户端登入了！

- 6. 修改安全性相关数据

由于我们建立的账号目录在 /winhome 底下，并非正规的 CentOS 目录，所以最重要的 SELinux 可能会跑掉～ 所以，我们还得要修订 SELinux 才行！方法很简单，将 SELinux type 转为 samba_share_t 即可！

```
[root@www ~]# chcon -R -t samba_share_t /winhome
```

由于 SELinux 的数据是会继承上层目录的，因此未来新增的用户，理论上，就不需要重新修订 SELinux 的文件类型了。 但是，如果你老是发现登入 PDC 的账号却无法取得家目录，那么就观察 /var/log/messages 内的资料来修订吧！

16.4.3 Wimdows XP pro. 的客户端

请注意，底下的方法仅适用于 Windows 2000, Windows XP 专业版 (Pro.)，一般的 Windows XP home 版本是不支持的！ 如果你客户端的主机是随机版的 Windows XP ，通常是 Windows XP home ，那底下的方法可能就无法适用啰！ 要连接上 Samba PDC 的过程也是挺简单的，你可以这样做： (至于 Windows 7 对于 Samba 的版本要求较高，官方网站是说得高于 3.3.x 以上版本才有支持)

- 1. 确认 windows 客户端的网域与主机名

首先我们必须要确认 Windows 客户端的工作组与主机名跟咱们的 Samba PDC 相同， 确认的方式在局域网络里面已经提过了，这里在强调一次。将鼠标移动到『我的计算机』上面，按下右键， 选择『内容』，然后点选『计算机名称』，会出现如下图示：

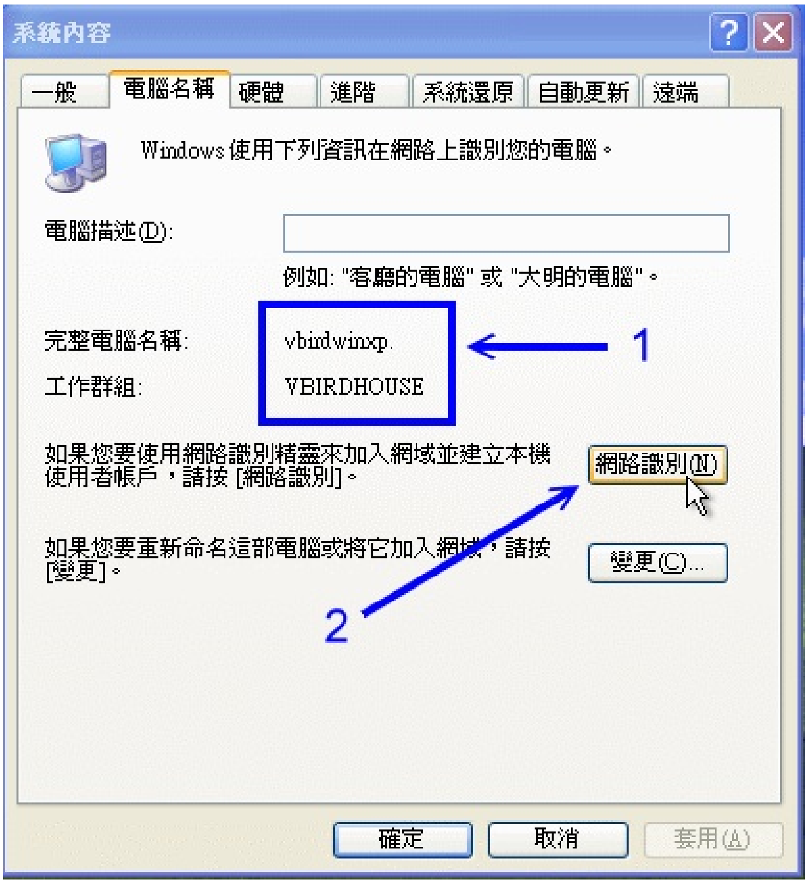


图 16.4-2、Windows 客户端连上 PDC 的方式流程示意图

如上图所示，你要先确认箭头 1 处指的主机名与工作组，在我们这个案例当中的工作组为 vbirdhouse， 这部 Windows 主机的 NetBIOS 名称则为 vbirdwinxp 喔！如果不对的话，请按下『变更』来设定， 并且重新启动。重新启动完毕后再到上图的画面当中，按下箭头 2 所指的网络识别处。

- 2. 设定主机名与域名

接下来我们要设定这部 Windows XP pro. 要链接到局域网络上的哪部 PDC 上面，亦即是处理主机账号以及 Samba PDC 负责的网域 (domain) 啦！在图 16.4-2 按下『网络识别』后，分别在出现的窗口当中选择：

1. 下一步；
2. 这台计算机是公司网络的一部份，而且我在工作时用来联机到其他计算机(T)
3. 我的公司使用一或多个网域的网络(C)
4. 下一步

然后就会出现如下的窗口：

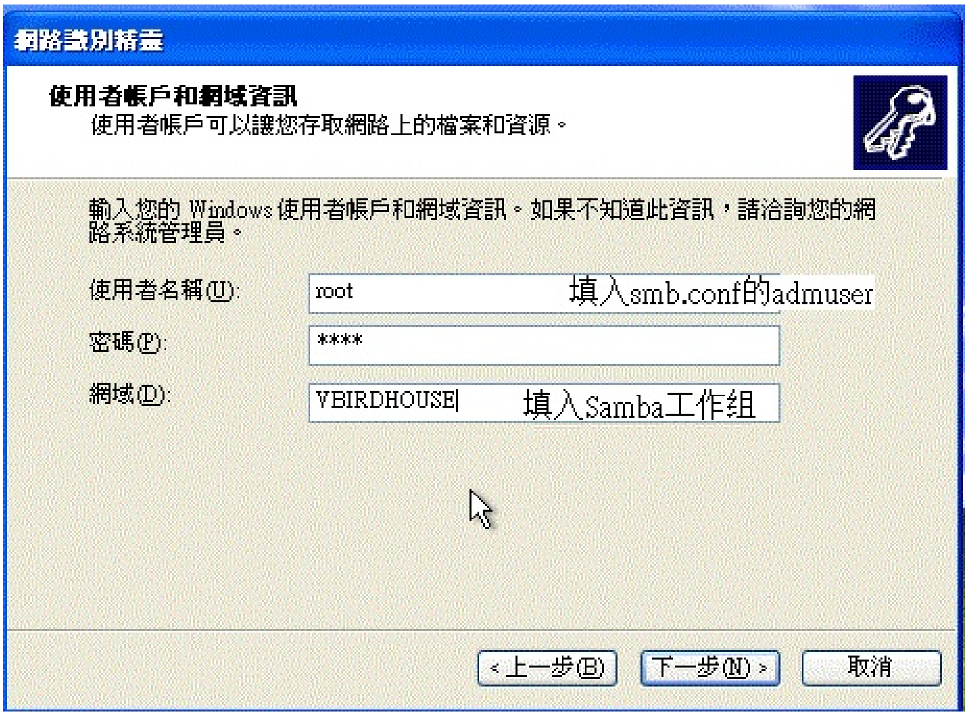


图 16.4-3、Windows 客户端连上 PDC 的方式流程示意图

请依序填写 Samba 主机上面的管理员账号与密码，要注意这个密码是记录于 Samba 中的那个，可不是 /etc/shadow 喔！别搞混了～这是 Samba 服务器的设定呢。输入之后按下一步吧，通常都会出现找不到正确主机的画面，如下所示：

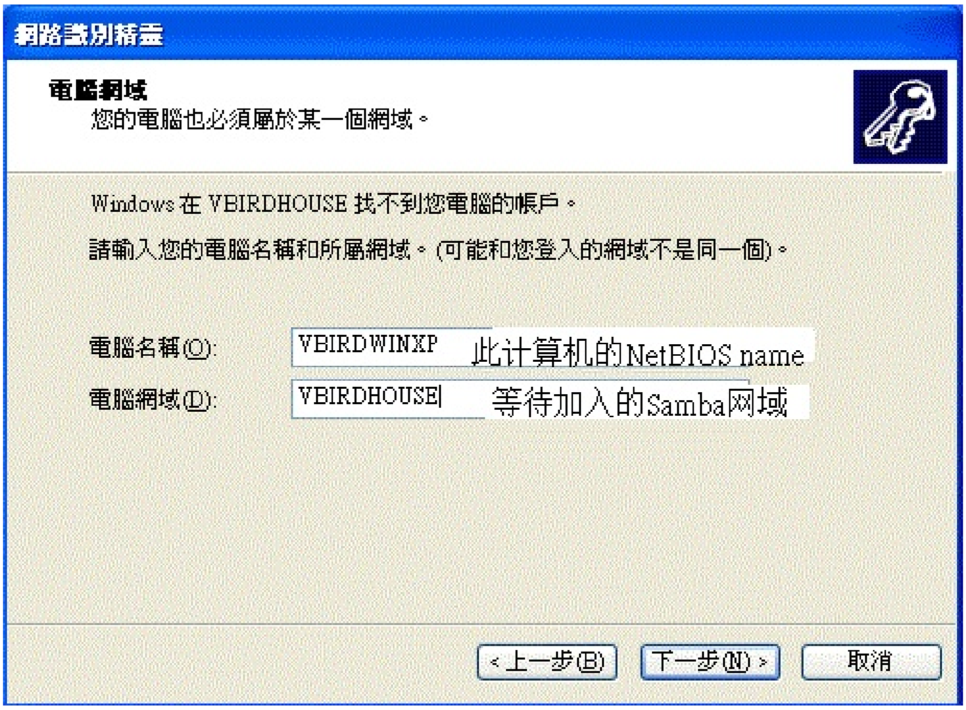


图 16.4-4、Windows 客户端连上 PDC 的方式流程示意图

鸟哥也觉得很奇怪，老是告诉我找不到！不过没有关系，这里我们依旧再填一次主机的 NetBIOS name 以及组名， 如上图所示，然后继续按下一步，就会出现如下的画面啦：



图 16.4-5、Windows 客户端连上 PDC 的方式流程示意图

这次就给他输入正确的管理员账号与密码，记得最后面的网域就是工作组名称，别写错了。处理完毕后给他按下确定吧！然后就会出现如下画面：

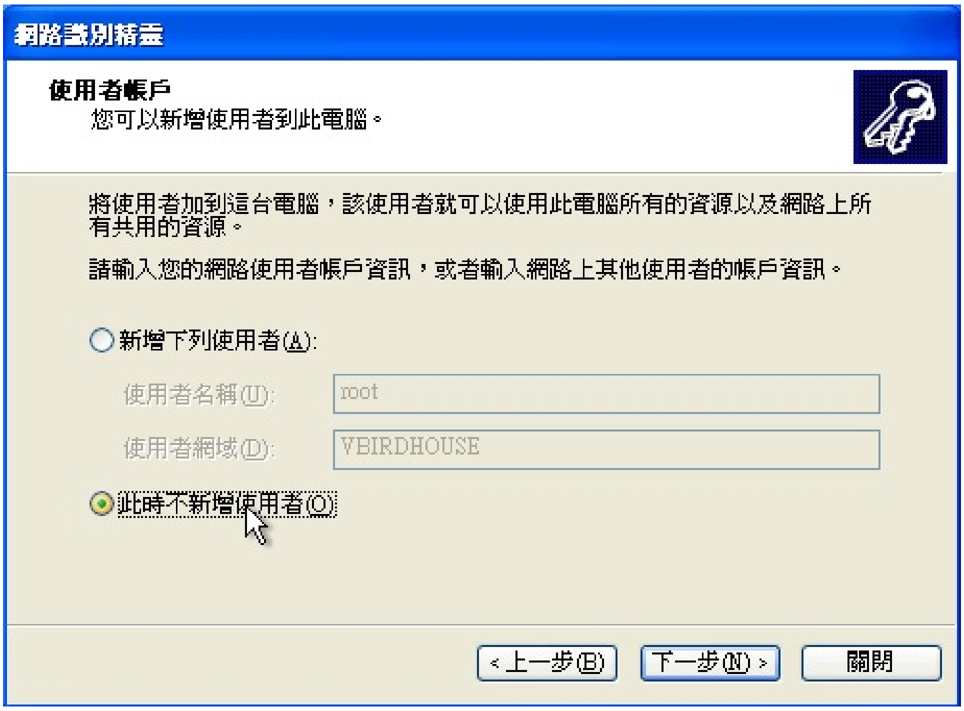


图 16.4-6、Windows 客户端连上 PDC 的方式流程示意图

恭喜你，这就表示已经连接上 Samba PDC 啰！我们希望所有的使用者都直接由 Samba PDC 控管，所以这里请填写『此时不新增使用者』吧！按下一步去。

- 3. 重新启动并以新的域名登入

在图 16.4-6 之后请重新启动，开机后整个画面会有点类似这样：

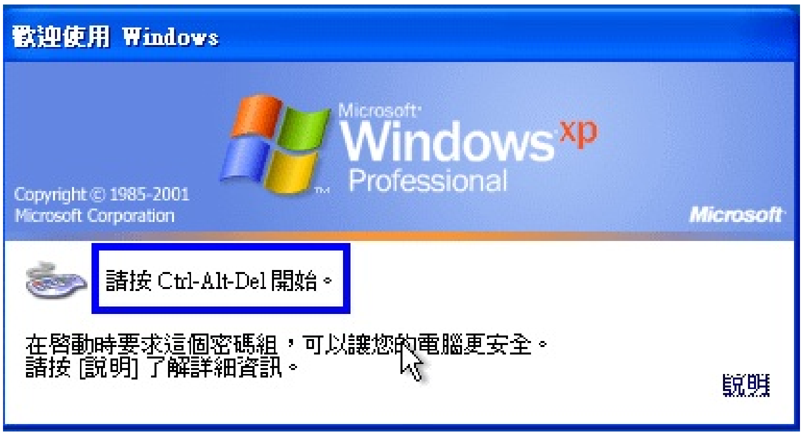


图 16.4-7、Windows 客户端连上 PDC 的方式流程示意图

为了保护我们的系统，因此得要按下 [ctrl]+[alt]+[del] 三个组合按键后，才会出现如下的登入画面：



图 16.4-8、Windows 客户端连上 PDC 的方式流程示意图

目前系统上面就会有两个可选择的账号管理模式，一个是本机账号一个是 PDC 提供的账号，那我怎知登入者是哪个管理模式？ 所以你就得要按下上述画面的『选项』，才会出现『登入到』的那一行数据。出现的两个数据分别是：

- VBIRDWINXP(此计算机)：这就是你的计算机名称，亦即是以本机账号登入；
- VBIRDHOUSE：就是 PDC 的 workgroup 项目，透过 PDC 的账号来尝试登入。

现在请输入你在 Samba PDC 上面拥有的账号与密码来尝试登入吧！那如果你输入的账号密码是对的，却发现如下的画面时， 肯定是某些档案权限或者是 SELinux 设定错误！请参考 /var/log/messages 或 /var/log/samba/* 里面的登录档来修改！

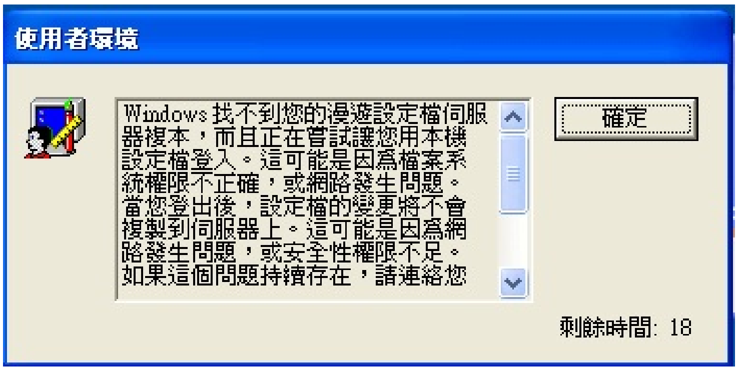


图 16.4-9、使用 PDC 账号登入却发现权限错误的图示

- 4. 观察用户的家目录与配置文件

如果你可以顺利登入的话，打开档案总管后应该可以看到类似下方的画面：

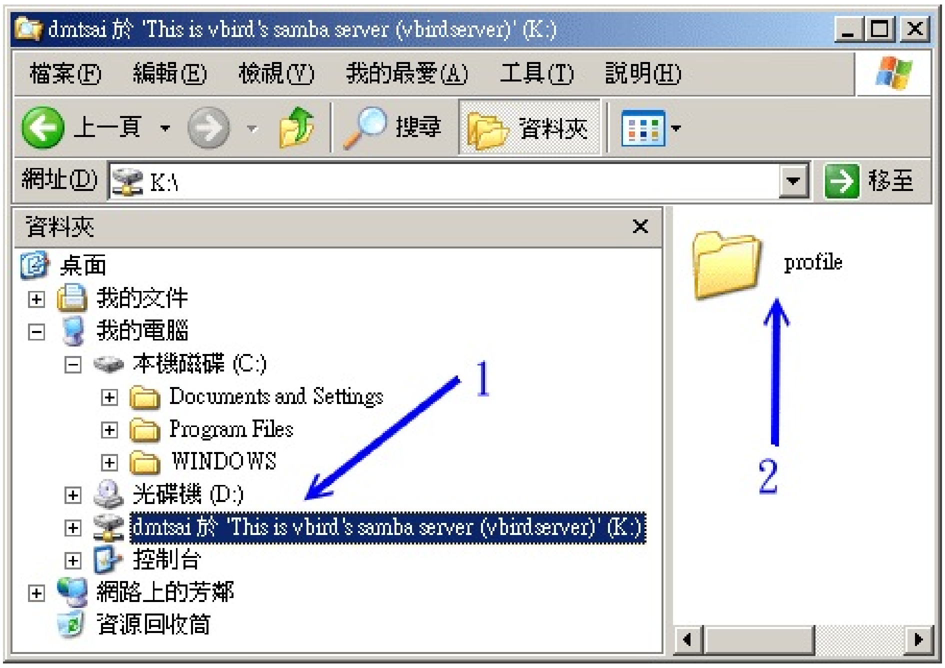


图 16.4-10、登入 PDC 后，取得的家目录状态

呵呵！该连上的通通连结上来啰！你也可以在自己的家目录 (K 槽) 新增移除数据的！是否很不错啊！ ^_^！ 而当你注销之后，你在 Windows 桌面上头所进行的各项个人化设定通通会被移动到 /winhome/dmtsai/profile 当中喔！ 如果不相信的话，请自行前往 Samba 服务器上头瞧一瞧就知道了。

16.4.4 Wimdows 7 的客户端

根据 SAMBA 官网的说明，支持 Windows 7 的 Samba 版本必须要高于 3.3.x 才行，还好，我们的 CentOS 6.x Samba 版本真的是高于 3.3.x 的 3.5.x，因此理论上是支持 Windows 7 的！只不过 Windows 7 要加入 Samba PDC 还得要修改注册码才行！这部份真的是给它很困扰！在 Windows 7 机码的修改方面，主要是修改底下的机码：

```
# 1\. 这个部分是进行『新增』机码！
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\LanmanWorkstation\Parameters]
"DomainCompatibilityMode"=dword:00000001
"DNSNameResolutionRequired"=dword:00000000
```

修改的方式为，在 Windows7 的执行里面输入『 regedit 』，会出现如下的画面：

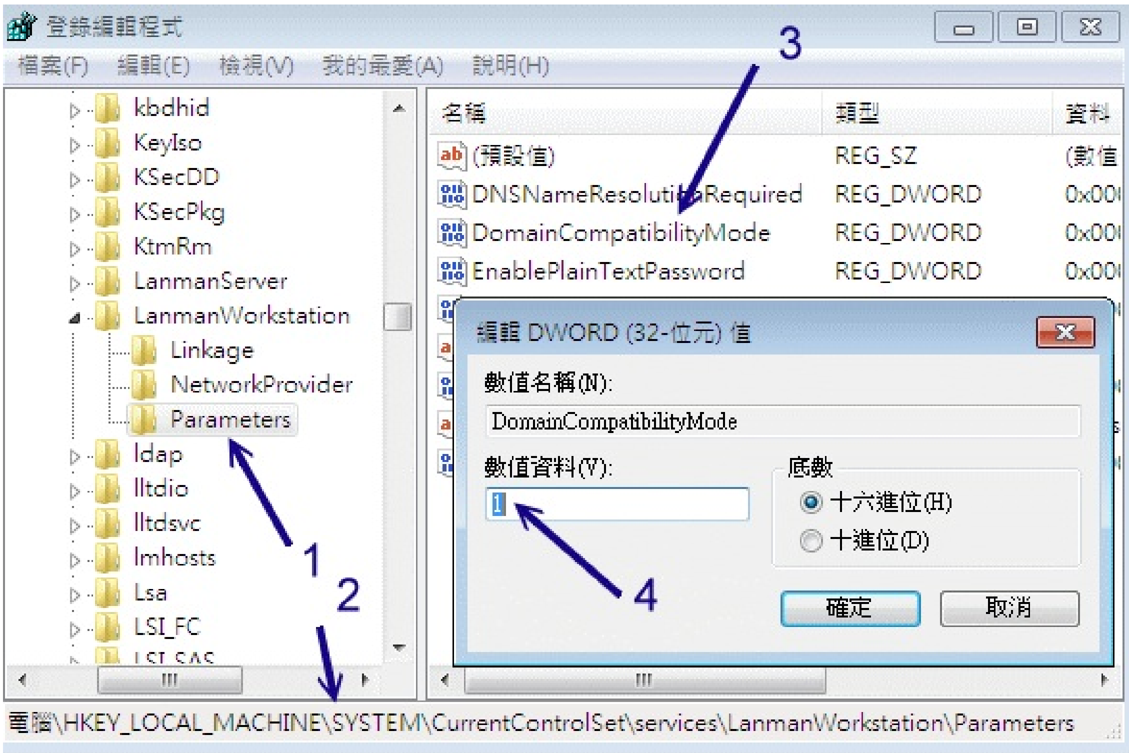


图 16.4-11、Windows 7 注册机码的动作

先由 (1)左侧窗口一层一层点选到我们所需要的目标去，然后 (2)观察最底下的机码顺序看对不对。之后 (3)在右侧窗口点选我们所需要的机码，如果是要新增，那就是在右侧空白处右键单击选新增即可增加一组机码名称。最后 (4)双击机码会出现可供修改的窗口，那就改成上面表格中的要求即可。更多关于 Windows 7 加入 PDC 的相关资料， 请查阅文末的参考数据部分喔。

等到将机码修订完毕，你就可以使用与 Windows XP 相同的方式来加入 PDC 啰！

16.4.5 PDC 之问题克服

如果老是发生错误讯息为『使用的帐户是计算机帐户。请使用你的通用用户帐户或本机用户帐户来存取这台服务器』时， 你可以这样做的：

- 先察看一下 /var/log/samba 里面的登录文件信息，尤其是 log.vbirdwinxp 关于这部主机的信息呐；

- 如果还是无法解决，可以在 lmhosts 里面增加 vbirdwinxp 的 IP 与主机名的对应，然后将 samba 整个关掉『/etc/init.d/smb stop』，等待一段时间让 NetBIOS 的名称解析时间逾时，再重新启动 samba 『/etc/init.d/smb start』，然后再重新做一次输入 root 的密码那个动作

在鸟哥尝试过的案例中，上面第二个步骤挺有效的！不过，还是得要察看 /var/log/samba 里面的登录信息才行喔！

- 一些 Windows 账号在 Windows 系统上面的使用技巧

虽然 PDC 很好用，不过你要注意的是，每次你使用 PDC 上头的账号登入 Windows 客户端主机时， Windows 主机会由 /winhome/username/profile/ 当中加载所需要的数据， 并暂时启动一个文件夹在 Windows 系统的 C:\Documents and Settings\username 当中，如果你的家目录下的 profile 数据太多时， 光是传输就会花去很多时间的！

所以，你应该将一些档案数据放置到你的家目录下，亦即 K 槽当中，尽量不要使用 Windows 预设的『我的文档夹』， 因为『我的文档夹』会将数据移动到『 /winhome/username/profile/My Documents/ 』目录下，同样的， 储存到桌面的数据会被放置到『 /winhome/username/profile/桌面/ 』目录中，那样在登入与注销时会花去很多时间喔！ 这个小地方也要注意的呢！ ^_^

好了，关于 SAMBA 的 PDC 作法我们就谈到这里，还有更多的信息你可以前往这个章节最后面的参考数据所列出的网址去查阅， 因为还有很多的作法呐！事实上，鸟哥觉得在一个网域当中，如果有多部的 Windows NT 主机，例如 Windows 2000/XP pro. 这一类的比较稳定的个人使用桌面版本时，使用 PDC 就很有用了！因为 Windows 2000/XP pro. 也是一个多人的操作系统，不像 Windows 98 是单人的操作系统。所以，当使用 Windows 2000/XP pro. 而无法登入 PDC 时，你是无法使用 Windows 2000/XP pro. 上面的任何的信息的。 但是在 Windows 98 上面若无法正确的登入，你仍然具有该计算机的主控权喔！

另外，设定 Windows 客户端之前，请先确认你的 Windows 是什么版本？ 上述的动作对于 Windows XP 家用版 (Home), Windows 7 是没有作用的！请先确认才行喔！

16.5 服务器简单维护与管理

除了上述的正规作法之外，其实还有一些稍微重要的事情要跟大家分享的！

16.5.1 服务器相关问题克服

通常我们在设定 SAMBA 的时候，如果是以单一主机的工作组 (Workgroup) 的方式来进行 smb.conf 的设定时，几乎很容易就可以设定成功了！并没有什么很困难的步骤。不过，万一还是无法成功的设定起来， 请务必察看登录档，也就是在 /var/log/samba/ 里面的数据！在这里面的资料当中，你会发现：咦！ 怎么这么多档案啊！因为我们在 smb.conf 里面设定了：

- log file = /var/log/samba/log.%m

那个 %m 是指客户端计算机的 NetBIOS Name 的意思，所以，当有个 vbirdwinxp 的主机来登入我们的 vbirdserver 主机时，那么登入的信息就会被纪录在 /var/log/samba/log.vbirdwinxp 档案喔！而如果万一来源 IP 并没有 Netbios name 的时候，那么很可能是一些错误讯息，这些错误讯息就会被纪录到 log.smbd, log.nmbd 里面去了！所以，如果你要察看某部计算机连上你的 SAMBA 主机发生了什么问题时，特别要留意这个登录档的形式喔！

另外，如果你的 SAMBA 明明已经启动完成了，却偏偏老是无法成功，又无法查出问题时，建议先关闭 Samba 一阵子，再重新启动：

- /etc/init.d/smb stop

在鸟哥过去的案例当中，确实有几次是因为 PID 与 NetBIOS 的问题，导致整个 SAMBA 怪怪的～所以完整的关闭之后， 经过一阵子的短暂时间，再重新启动，应该就可以恢复正常了！

还有，万一你在进行写入的动作时，老是发现『你没有相关写入的权限！』，不要怀疑，几乎可以确定是 Permission 的问题，也就是 Linux 的权限与 SAMBA 开放的权限并不相符合，或者是 SELinux 在搞鬼！无论如何， 你必须要了解能不能写入 Linux 磁盘，看的是 PID 的权限与 Linxu 文件系统是否吻合，而那个 smb.conf 里面设定的相关权限只是在 SAMBA 运作过程当中『预计』要给使用者的权限而已，并不能取代眞正的 Linux 权限喔！所以，万一眞的发现该问题存在， 请登入 Linux 系统，查验一下该对应的目录的 permission 吧！^_^

另外，通常造成明明已经查到分享 (smbclient -L 的结果)，却老是无法顺利挂载的情况，主要有底下几个可能的原因：

- 虽然 smb.conf 设定正确，但是设定值『 path 』所指定的目录却忘记建立了 (最常见的呆样！)；
- 虽然 smb.conf 设定为可擦写，但是目录针对该用户的权限却是只读或者是无权限；
- 虽然权限全部都正确，但是 SELinux 的类型却错误了！
- 虽然全部的数据都是正确的，但是 SELinux 的规则 (getsebool -a) 却没有顺利启动。

上述都是一些常见的问题，更多问题的解决方案，请参考最正确的登录文件信息吧！ ^_^

16.5.2 让使用者修改 samba 密码同时同步更新 /etc/shadow 密码

有个问题是，我们知道使用者可以透过 passwd 修改 /etc/shadow 内的密码，而且用户也能够自行以 smbpasswd 修改 Samba 的密码。如果用户是类似 PDC 的用户，那么这些用户理论上就很少使用 Linux 啦！那么想一想， 能否让用户在修改 Windows 密码 (就是 Samba) 时，同步更新 Linux 上面的 /etc/shadow 密码呢？ 答案是可行的啦！而且动作并不困难～因为 smb.conf 里头已经提供了相对应的参数设定值！你可以参考底下的网站数据：

- http://moto.debian.org.tw/viewtopic.php?t=7732&
- http://de.samba.org/samba/docs/using_samba/ch09.html

鸟哥做个总结，基本上你需要的是 smb.conf 里面 [global] 的几个设定值：

```
[root@www ~]# vim /etc/samba/smb.conf
[global]
# 保留前面的各项设定值，并新增底下三行即可：
 unix password sync  = yes                &lt;==让 Samba 与 Linux 密码同步
        passwd program      = /usr/bin/passwd %u &lt;==以 root 呼叫修改密码的指令
        pam password change = yes                &lt;==并且支持 pam 模块！

[root@www ~]# testparm
[root@www ~]# /etc/init.d/smb restart
```

接下来，当你以一般用户 (例如 dmtsai) 修改 samba 的密码时，就会像这样：

```
[dmtsai@www ~]$ smbpasswd
Old SMB password:  &lt;==得先输入旧密码，才能输入新密码
New SMB password:
Retype new SMB password:
Password changed for user dmtsai &lt;==这就是成功的字样！

# 若出现底下的字样，应该就是你的密码输入被限制了！例如输入的密码字符少于 6 个！
machine 127.0.0.1 rejected the password change: Error was : Password restriction.
Failed to change password for dmtsai
```

16.5.3 利用 ACL 配合单一使用者时的控管

想象一个案例，如果你是学校的网管人员，有个兼任老师向你申请账号，主要是要在很多班级内取得同学的专题资料。 因为该老师是兼任的，你或许担心一不小心该教师就将同学的辛苦资料给销毁，倒不是教师们故意的， 而是很多时候...不熟嘛！这个时候如果你将该老师加入同学的群组，然后偏偏同学们所在的目录是群组可写入的话， 那么该教师就能够拥有可擦写的权限了，也就容易造成一些莫名的灾难～

那该怎么办？其实可以透过 ACL 来管理某个目录的单一用户权力啦！所以说， 权限的管理不必透过 smb.conf 的设定，只要透过 ACL 来管理就能够达到你所需要的目的了。 关于 ACL 的说明我们在基础学习篇第三版第十四章已经提过了， 这里不再啰唆，请自行前往查阅呐！ ^_^

16.6 重点回顾

- 由 Tridgell 利用逆向工程分析网芳得到 Server Message Block 协议的产生；
- Samba 名称的由来是因为需包含没有意义的 SMB server 之故；
- SAMBA 可以让 Linux 与 Windows 直接进行文件系统的使用；
- SAMBA 主要架构在 NetBIOS 上发展的，且以 NetBIOS over TCP/IP 克服 NetBIOS 无法跨路由的问题；
- Samba 使用的 daemon 主要有管理分享权限的 smbd 以及 NetBIOS 解析的 nmbd
- Samba 使用的模式主要有单机的 workgroup 方式，以及网域控管的 PDC 模式；
- Samba 的主配置文件之档名为 smb.conf
- smb.conf 内，主要区分为 [global] 服务器整体设定与 [share] 分享的资源两大部分
- Samba 使用者账号控管主要的设定值为 security = {share,user,domain} 等
- Samba 客户端可使用 smbclient 以及 mount.cifs 进行网芳的挂载
- 新版的 Samba 默认使用数据库记录帐户信息，新增账号用 pdbedit ，修改密码则用 smbpasswd
- Samba 主要支持 CUPS 的打印机服务器
- 在权限控管方面，最容易出错的为 SELinux 的规则与类型 (SELinux type)
- 在 PDC 的设定方面，由于与主机名相关性很高，建议设定 lmhosts 档案内容为宜

16.7 本章习题

- 一般来说， SAMBA 使用的配置文件放置在哪里？档名为何？使用的档名为 smb.conf ，通常会放置在 /etc/samba/smb.conf 里面，不过，最好可以使用 rpm -qc packagename 来查询！
- 哪一个指令可以用来判断 smb.conf 这个配置文件的正确性？当我们修改完 smb.conf 之后，记得要以 testparm 来进行 samba 的确认！
- 哪一个指令可以用来察看 SAMBA 主机分享出什么目录？利用 smbclient 即可：『smbclient -L NetBiosName -U username 』！
- 在 Linux 客户端挂载网芳的文件系统主要是依据哪个指令来达成的？就是透过 mount.cifs 或 mount -t cifs 来达成的！
- 我今天使用 smbpasswd 去新增一位使用者 badbird，让他可以登入我的 Linux SAMBA 主机，但是无论如何就是无法新增。你认为原因可能是什么？由于 Samba 用户的信息必须要存在于 /etc/passwd 里面，既然无法新增，应该先确认 badbird 这个用户已经存在于 Linux 系统当中了！

16.8 参考数据与延伸阅读

- 注1：维基百科对 Samba 的来源与作者的介绍：http://en.wikipedia.org/wiki/Samba_software http://en.wikipedia.org/wiki/Andrew_Tridgell
- man 5 smb.conf
- Study Area ：http://www.study-area.org/linux/servers/linux_samba.htm
- 电子书 Using Samba：http://de.samba.org/samba/docs/using_samba/ch00.html
- Samba PDC HOWTO: http://us5.samba.org/samba/docs/man/Samba-HOWTO-Collection/samba-pdc.html
- SAMBA 官方网站：http://www.samba.org/
- 杨锦昌老师的 SAMBA 密技：http://apt.nc.hcc.edu.tw/web/student_server_FC1.htm#samba
- 依玛猫的打印文件：http://www.imacat.idv.tw/tech/lnxprint.html
- Gentoo Linux 的 Samba 文件：http://www.gentoo.org/doc/zh_tw/quick-samba-howto.xml
- cupsaddsmb 用法：http://www.enterprisenetworkingplanet.com/netsysm/article.php/3621876
- 下载 CUPS-windows 的网站：http://ftp.easysw.com/pub/cups/windows/
- eyesblue 在讨论区针对语系的说明：http://phorum.vbird.org/viewtopic.php?t=22001
- testparm -v
- 关于 Windows 7 加入 PDC 的机码相关问题：https://wiki.samba.org/index.php/Windows7 http://www.linuxquestions.org/questions/linux-server-73/joining-a-windows-7-client-to-samba-pdc-v-3-4-3-a-815174/ http://www.1stbyte.com/2009/05/31/join-windows-7-to-samba-pdc/

```
# 2\. 这个部分是进行『修改』机码！不过，如果你作了，会让你可加入 PDC，
# 但却无法顺利的登入！所以这里得要特别注意！
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\Netlogon\Parameters]
“RequireSignOnSeal”=dword:00000000
“RequireStrongKey”=dword:00000000
```

2001/09/17：好久以前曾经完成的一项任务 ^_^ 2003/07/26：将 2001/09/17 所写的内容做了大幅度的修订，增加原理以及更多的设定项目！ 2003/09/10：将 PDC 部分补充的更完整，因为加入了个人化的 Profiles 在 /home/samba/profiles 当中了！同时加入课后练习喔 2003/09/30：加入了 CUPS 打印机的支持！ 2005/10/17：Samba 2.2 在中文编码上面与最新的 samba 3.0.x 版本不同。请参考： 中文编码 网友的详细说明喔！ 2006/12/20：将旧的文章移动到 此处 2006/12/29：终于写完了 Samba 了！将 PDC 改写，很多乱乱的地方都改掉了～ 2007/04/12：原本对 homes 的说明中，那个 umask 应该是 002 ，原先的 022 是错的！

2010/06/11：一直误会了作者的名字，名称为 Andrew Tridgell 而不是 Tridgwell ！抱歉了！ 2011/03/18：将旧的基于 CentOS 4.x 的文章移动到 此处 2011/03/31：终于搞定了 Samba ，光是实作就花去鸟哥大部分的时间了～累毙！ 2011/07/29：将基于 CentOS 5.x 的版本移动到 此处 2011/07/29：PDC 的部分，终于可以加入 windows 7 啰！开心！

第十七章、区网控制者： Proxy 服务器

最近更新日期：2011/08/02

代理服务器的功能是可以代理局域网络的个人计算机来向因特网取得网页或其他数据的一种服务， 由于代理取得的数据可以保存一份在服务器的快取上，因此以往有类似『假象加速』的功能！不过，目前网络带宽已经比以前好很多， 因此代理服务器倒是很少使用在这方面。取而代之的是局域网络『高阶防火墙』的角色！这里的『高阶』指的是 OSI 七层协议里面的高层，因为代理服务器是用在应用层上的一种防火墙方式啦！不像 iptables 是用在网络、传输层。Linux 上启动代理服务器的是 squid 这个软件呦！

17.1 什么是代理服务器 (Proxy)

代理服务器 (Proxy) 的原理其实很简单啦！就是以类似代理人的身份去取得用户所需要的数据就是了！ 但是由于它的『代理』能力，使得我们可以透过代理服务器来达成防火墙功能与用户浏览数据的分析！ 此外，也可以藉由代理服务器来达成节省带宽的目的，以及加快内部网络对因特网的 WWW 访问速度！总之， 代理服务器对于企业来说，实在是一个很不错的东西啊！

17.1.1 什么是代理服务器

在真实世界中，我们或许会帮忙家人去办理一些杂务吧！举个例子来说，例如缴费或者是申办提款卡等等的， 由于你并不是『申请者本人』而是『代理人』的角色，因此有时候会需要秀出一些证件就是了。 那么在网络上面的代理服务器 (Proxy Server) 是怎么回事呢？它最主要的功能就如同我们上面提的真实世界一样， 当客户端有因特网的数据要求时，Proxy 会帮用户去向目的地取得用户所需要的数据。 所以，当客户端指定 WWW 的代理服务器之后，用户的所有 WWW 相关要求就会通过代理服务器去捉取啰！ 整个代理服务器与客户端的相关性可以由下图约略看出一个端倪：

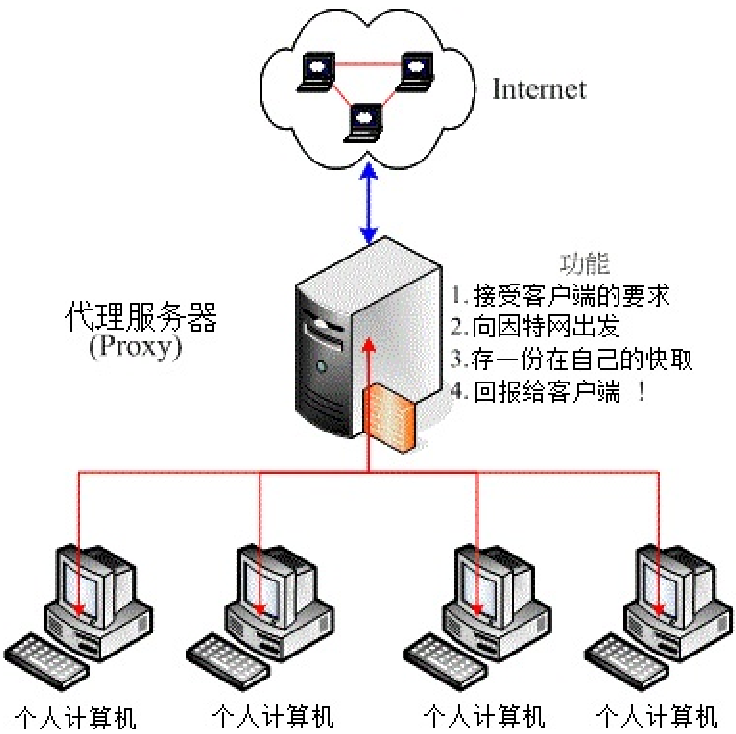


图 17.1-1、代理服务器、客户端与因特网的相关性示意图

一般来说，代理服务器会架设在整个区网的单点对外防火墙上头，而在区网内部的计算机就都是透过 Proxy 来向因特网要求数据的，这就是所谓的『代理服务器』啦！当然，上面的架构仅只是一个案例，但是这个架构比较多人用的原因， 是因为这样的 Proxy server 还可以兼

做高阶防火墙之用啦！

在 Proxy 与客户端的相关性当中，你必需要了解的是：客户端向外部要求的资料事实上都是 Proxy 帮用户取得的，因此因特网上面看到要求数据者，将会是 Proxy 服务器的 IP 而不是客户端的 IP。 举个例子来说，假如鸟哥在我的浏览器设定了我们学校的代理服务器主机 proxy.ksu.edu.tw 做为我的 Proxy 好了，再假设我的 IP 是 120.114.141.51 ，那么当我想要取得 Yahoo 的新闻信息时，事实上，都是 proxy.ksu.edu.tw 帮我去取得的，所以在 Yahoo 的网站上面看到要求数据的人是谁呢？呵呵！当然就是 proxy.ksu.edu.tw 而不是 120.114.141.51 啰！这样可以了解 Proxy 的功能了吗？

除了这个功能之外，Proxy 还有一个很棒的额外功能，那就是防火墙的功能！ 看一下上面的图示，你可以发现一件事情，那就是客户端的个人计算机要连上因特网一定要经过 Proxy 服务器。并且，如果有人想要入侵你的系统时，由于你的 proxy 在最外部啊，所以攻击者就会攻击错方向，如此一来，不就比较安全！ 此外，由于整个因特网对外都是经过 proxy ，也就是『单点对外』的情况，这种状态底下要来管理防火墙也是比较简单的喔！^_^

17.1.2 代理服务器的运作流程

了解了 Proxy 的功能之后，我们来谈一谈那么 Proxy 到底是怎样运作的呢？为何它会有『加快网络存取效率』的好处？ 这就必需要以底下的图示来说明了！

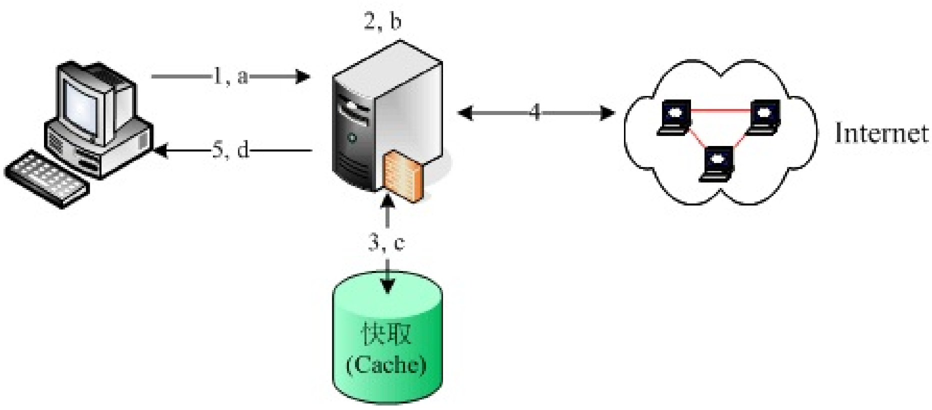


图 17.1-2、代理服务器的运作流程图：快取数据与客户端

当客户端指定了代理服务器之后，在客户端想要取得因特网上面的信息时，它是这样取得数据的 (注：那个 Cache 表示为 Proxy 服务器的硬盘的意思)：

- 当 Proxy 的快取拥有用户所想要的数据时 (Step a ~ d)：
- Client 端向 Server 端发送一个数据需求封包；
- Server 端接收之后，先比对这个封包的『来源』与预计要前往的『目标』网站是否为可接受？ 如果来源与目标都是合法的，或者说，来源与目标网站我们的 Proxy 都能帮忙取得资料时，那么 Server 端会开始替 Client 取得资料。这个步骤中比较重要的就是『比对政策』啦，有点像是认证的感觉啦；

- Server 首先会检查自己快取 (新的数据可能在内存中，较旧的数据则放置在硬盘上) 数据， 如果有 Client 所需的数据，那就将数据准备取出，而不经过向 Internet 要求数据的程序；
- 最后当然就是将数据回传给 Client 端啰！
- 当 Proxy 的快取没有用户所想要的数据时 (Step 1 ~ 5)：
- Client 端向 Server 端发送一个数据需求封包；
- Server 端接收之后，开始进行政策比对；
- Server 发现快取并没有 Client 所需要的资料，准备前往因特网抓取数据；
- Server 开始向 Internet 发送要求与取得相关资料；
- 最后当然就是将数据回传给 Client 端啰！

上面的流程分析里面，我们可以清楚的知道，当 Proxy 曾经帮某位用户取得过 A 数据后，当后来的用户想要重复取得 A 数据时，那么 Proxy 就会从自己的快取里面将 A 数据取出传送给用户，而不用跑到因特网去取得同样的这份资料喔。因为没有去因特网找数据，当步骤 4 的流程很花时间时，那么透过 Proxy 忽略步骤 4 ，感觉上就好像网络速度变快了！但其实只是直接从 Proxy 的快取里面抓而已 (所以才会有人说『假象网络加速』的功能)！这就是两个流程最大的差异了。

在目前的因特网社会里，由于宽带技术已经很成熟，所以在不乱用的情况下，网络带宽理论上是足够的 (除非要连到国外去)。 那么用了 Proxy 之后效能会不会更提升呢？答案是，『应该不会』！啥？怎么会这样呢？从上面的流程分析中， 我们发现 Proxy 会常常去读取硬盘内的数据，而硬盘内的快取数据又是透过某些特殊方式在管理， 因此要找到该份数据就要花一些时间，再加上如果硬件效能 (硬盘或主板芯片组) 不佳时，那么加了 Proxy 反而会让你感觉网络传输怎么『卡卡的』呦！这点得要特别注意才行！

Tips: Proxy 对于 cache 的速度是很要求的，而这个 cache 就是硬盘啦！当然，硬盘容量必需要足够大，而且还要『足够快』才行！ 因为由上面的流程当中，我们不难发现，cache 是一直被重复存取的一个地方喔！所以硬盘的好坏就差别很大啦！可以说他是影响一个 Proxy 效能好坏的关键点呢！



17.1.3 上层代理服务器

想一想，既然 Proxy 是帮忙客户端进行网页代理的工作，那么我们的 Proxy 能不能也指定另外一台 Proxy 当成我的 Proxy 的 Proxy 呢？很绕口吧！其实流程像底下这样啦：

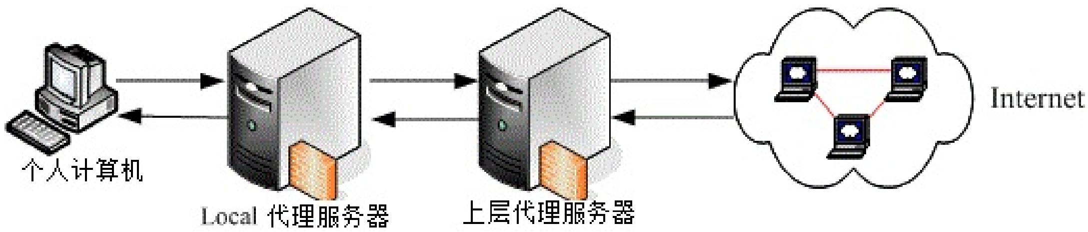


图 17.1-3、上层代理服务器示意图

就是我们的 Local proxy 并不会主动的去捉数据，而是再透过『上层代理服务器』向 Internet 要求资料！这样有什么好处呢？由于可做为我们的上层代理服务器的主机通常是具有较高带宽的， 因此我们透过它去要求数据当然『理论上』速度会更快喔！而上层代理服务器最大的好处其实是在于『分流』喔！ 例如下图所示：

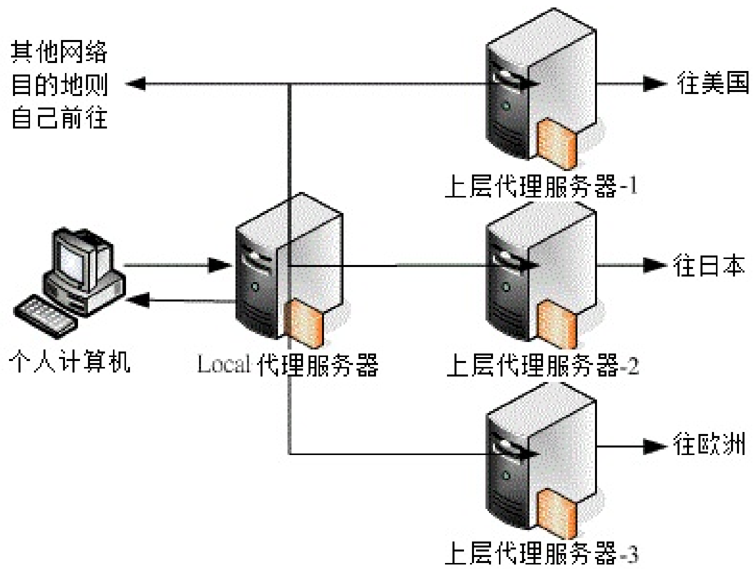


图 17.1-4、以多部上层代理服务器达到分流的效果示意图

我总共设定了三部上层代理服务器，由于这三个代理服务器对外的速度都不相同，所以，当我要去美国时，就以 Proxy1 来要求资料，要连欧洲就以 Proxy3 ，至于要连日本，就以 Proxy 2 来要求我所需要的数据，如此一来，呵呵！可以让我的 Proxy 达到最佳的效能喔！很不错吧！此外，为了节省上层 proxy 的负担，如果是其他网络位置，我们则设定由自己的 local proxy 捉取～ 设定的弹性很高呢！

由于代理服务器需要管控信任的来源端客户端计算机，因此各 ISP 仅能针对自家的用户来开放 Proxy 使用权而已。 台湾常见的几家 ISP 提供的 Proxy 有：

- Hinet： http://service.hinet.net/2004/new_adsl04.htm
- SeedNet： https://service.seed.net.tw/home/setting/server.htm

由于当用户透过 Proxy 连到因特网时，网络看到的是 Proxy 在抓取数据而不是该客户端，因此，我们不难发现 Proxy 有可能会被客户端过度的滥用，同时也有可能会被拿来为非作歹啊！所以，目前绝大部分的 Proxy 已经『停止对外开放』了，仅针对自己的网域内的用户提

供本项服务而已～

因此，如果你要自行设定 Proxy 的时候，请记得去你当初申请网络的 ISP (如果是学术单位，请到贵单位的计中网页瞧瞧即可) 搜寻一下，才能比较有效的设定好你的服务器喔！因为设定错误的话，呵呵！上层 Proxy 根本不提供服务，或者是上层 Proxy 的效能并不好，那个时候你的 Proxy 也会连带的受到很大的影响啊！慎选！慎选！

17.1.4 代理服务器与 NAT 服务器的差异

或许你已经发现了一件事，那就是：在内部局域网络使用私有 IP 的客户端，不论透过 Proxy 或者 NAT 均可以直接取得 WWW 的服务，那么 NAT 与 Proxy 有没有什么不同的地方啊？它们不都是可以让内部的计算机连接出去吗？其实这两个玩意儿差异性是『相当大』的喔！ 简单说明如下：

- NAT 服务器的功能： 就如同第九章提到的数据，Linux 的 NAT 功能主要透过封包过滤的方式， 并使用 iptables 的 nat 表格进行 IP 伪装 (SNAT) ，让客户端自行前往因特网上的任何地方的一种方式。主要的运作行为是在 OSI 七层协定的二、三、四层。由于是透过封包过滤与伪装，因此客户端可以使用的端口号码 (第四层) 较弹性；
- Proxy 服务器的功能： 主要透过 Proxy 的服务程序 (daemon) 提供网络代理的任务，因此 Proxy 能不能进行某些工作，与该服务的程序功能有关。 举例来说，如果你的 Proxy 并没有提供邮件或 FTP 代理，那么你的客户端就是无法透过 Proxy 去取得这些网络资源。 主要运作的行为在 OSI 七层协议的应用层部分 (所谓的比较"高阶"之意)。

这样说有没有比较有点概念了呢？NAT 服务器是由较底层的网络去进行分析的工作，至于通过 NAT 的封包是干嘛用的， NAT 不去管他！至于 proxy 则主要是由一个 daemon 的功能达成的，所以必需要符合该 daemon 的需求，才能达到某些功能！

17.1.5 架设代理服务器的用途与优缺点

现在我们约略知道 Proxy 的功能了，那么通常什么情况下会架设 Proxy 呢？一般来说，代理服务器的功能主要有：

- 作为 WWW 的网页资料取得代理人：这是最主要的功能嘛！
- 作为内部区网的单点对外防火墙系统：如图 17.1-1 所示一般，如果你的 Proxy 是放在内部区网的 Gateway 上头，那么这部代理服务器就能够作为内部计算机的防火墙了！而且还不需要设定那复杂的 NAT 功能呢！只是单纯的 Proxy 服务器通常仅提供 WWW 的代理，因此内部计算机想要取得 smtp, ftp...就比较麻烦～

由于 Proxy 的这种特性，让他很常被使用于大型的企业内部，因为可以达到杜绝内部人员上班时使用非 WWW 以外的网络服务，而且还可以监测用户的资料要求流向与流量呢！很不错吧！ ^_^！好了，接下来我们来谈一谈当你架设了 Proxy 后的优缺点吧。先来谈谈主要可能具有的优点有：

- 节省单点对外的网络带宽，降低网络负载：当你的 Proxy 用户很多时，那么 Proxy 内部的快取数据将会累积较多。因此客户端想要取得网络上的数据时，很多将会从 Proxy 的快取中取得，而不用向因特网要求资料。 所以可以节省带宽啊！
- 以较短的路径取得网络数据，有网络加速的感觉：例如你可以指定你的 ISP 提供的代理服务器连接到国外，由于 ISP 提供的 Proxy 通常具有较大的对外带宽，因此在对国外网站的数据取得上， 通常会比你自己的主机联机到国外要快的多。此外，与上一点的快取数据也有关系啊！从内部硬盘取得的路径总比对外的因特网要短的多啊！
- 透过上层代理服务器的辅助，达到自动数据分流的效果：例如图 17.1-4 所示，让客户端在不知不觉之间，就可以得到数据由不同 Proxy 取得的加速效果！
- 提供防火墙内部的计算机连上 Internet：就是上面提到的单点对外防火墙功能！

由于代理服务器的这些优点，因此这里要强烈的建议，如果你需要连上国外的网页， 请一定使用 ISP 提供给你的代理服务器来帮忙，因为不但可以节省带宽，并且速度上会快上很多很多 (例如美国环保署, EPA 网站)。 不过，有利就有弊，当然 Proxy 也不是万能的天神～他有什么可能潜藏的缺点呢？

- 容易被内部区网的人员滥用：我们知道因特网上看到取得资料的人是 Proxy 那部主机而不是客户端计算机的 IP，因此可能会让某些内部网络使用人员开始利用你的 proxy 干坏事，此时你就会很麻烦～ 所以，为了杜绝这个状况，强烈的建议多加登录档案分析的软件，在管理上面会轻松很多喔！
- 需要较高超的设定技巧与除错程序：在鸟哥设定过的服务器当中， Proxy 算是比较不容易设定好『效能』的一个服务器了！由于 Proxy 的 Cache 与他的『上层代理服务器』的关系是很紧密的， 万一设定错误的话，很有可能反而让你的 Proxy 拖垮客户端 WWW 的浏览速度！最严重的是造成无法联机！
- 可能会取得旧的错误数据：这个最容易发生了！由于曾经浏览过的网页会被放置到快取， 并提供后续用户的直接取得。万一因特网上面的那个网页数据更新过呢？那时你会发现，怎么客户端无法看到更新后的资料？ 就是因为快取的问题啊！取得旧数据的频率可能会很高啊！

总之， Proxy 的优点是很多的，但是缺点却需要网管人员的操心啊！既然如此，那么我们到底有没有需要架设代理服务器呢？ 简单的说，我们可以这样分析！

- 我的 Client 端用户不少，而且大部分仅需要 WWW 这个网络服务而已；
- 我的 Proxy 还兼做防火墙的任务；
- 我的 Client 端常常需要联机到传输速度很慢的网站，例如国外的网站；

- 我的 Client 端常常浏览的网站是『静态』网站，而不是动态网站 (例如讨论区的 PHP)。

如果你有上述的环境状况，那么是可以考虑架设 Proxy 的，但是，相反的来说，要是 (1)我的 Client 端很少，所以每次连上 WWW 都是求取新的资料 (并没有用到快取)，有没有 Proxy 反而看不出效益～此外，(2)Proxy 由于属于应用层了，对于 Internet 的规划上弹性较不足！不像 NAT 服务器可以进行很多的功能！(3)我常常上的网站是类似讨论区那种一日多变的网站，在这样的情况下，实在是没有必要架设 Proxy 的！

但是，如果对于学校单位那原本带宽就不足的环境中，架设 Proxy 来让校内的网络速度提升，呵呵！就是有那个必要性的啦！所以要不要架设 Proxy 呢？请好好的依据你的环境来考虑喔！但无论如何，我们还是要教大家怎么架设它就是了 ^_^

17.2 Proxy 服务器的基础设定

虽然在我们小型的网络环境中，架设 Proxy 真的没有什么用，不过，考虑到大家未来可能会高升嘛！所以企业常用的 Proxy 也需要了解一下比较好。 在这个小节中，我们主要介绍一个比较简单的 Proxy 环境，就是单纯可以跑而已的代理服务器。比较高阶的设定请参考后续小节的介绍啰。

17.2.1 Proxy 所需的 squid 软件及其软件结构

达成代理服务器功能的软件很多，例如效能不是很好的 Apache 以及我们这个章节要介绍的八爪章鱼 squid 这一套。 目前代理服务器在 Unix Like 的环境下，大多就是使用 squid ，因此我们这里以 squid 为准来介绍啦。同样的， 请使用 rpm 来检查，如果尚未安装，请用『 yum install squid 』来安装吧！安装好 squid 之后，它主要的提供的配置文件有：

- /etc/squid/squid.conf 这个是主要的配置文件，所有 squid 所需要的设定都是放置在这个档案当中的！ 鸟哥底下提到的种种设定方法几乎都是这个档案里面的说明喔！

- /etc/squid/mime.conf 这个档案则是在设定 squid 所支持的 Internet 上面的文件格式，就是所谓的 mime 格式啰！ 一般来说，这个档案的预设内容已经能够符合我们的需求了，所以不需要更动他，除非你很清楚的知道你所需要额外支持的 mime 文件格式。

其他重要的目录与档案有：

- /usr/sbin/squid：提供 squid 的主程序啊！
- /var/spool/squid：就是默认的 squid 快取放置的目录。
- /usr/lib64/squid/：提供 squid 额外的控制模块，尤其是影响认证密码方面的程序，都是放在这个目录下的；

17.2.2 CentOS 预设的 squid 设定

在预设的情况下，CentOS 的 squid 具有底下几个特色：

- 仅有本机 (localhost, 127.0.0.1) 来源可以使用这个 squid 功能
- squid 所监听的 Proxy 服务埠口在 port 3128
- 快取目录所在的位置在 /var/spool/squid/ ，且仅有 100MB 的磁盘高速缓存量
- 除了 squid 程序所需要的基本内存之外，尚提供 8MB 的内存来给热门档案快取在内存中 (因为内存速度比硬盘还快)
- 默认启动 squid 程序的用户为 squid 这个账号 (与磁盘高速缓存目录权限有关)

其实， CentOS 预设的 squid 设定，是仅针对本机 (localhost) 开放的情况，而一大堆设定的默认值， 都是仅针对小型网络环境所指定的数值，同时，很多比较特殊的参数都没有启动。所以，我们就得要来了解一下各设定值的意义， 这样才能够进行修改嘛！这些参数都是在 squid.conf 里头指定的，所以，就让我们来看看这个档案的内容与较重要的参数吧：

Tips: CentOS 6.x 已经将 squid.conf 里面不相干的设定值通通拿掉了，所以这个档案就变的非常的精简！这样其实有好有坏啦！ 好处是，你不用去看一些你用不到的参数值，坏处是，如果你想要其他的设定，就得额外参考外部文件了！伤脑筋～



```
[root@www ~]# vim /etc/squid/squid.conf
# 1\. 信任用户与目标控制，透过 acl 定义出 localhost 等相关用户
acl manager proto cache_object              &lt;==定义 manager 为管理功能
acl localhost src 127.0.0.1/32              &lt;==定义 localhost 为本机来源
acl localhost src ::1/128
acl to_localhost dst 127.0.0.0/8 0.0.0.0/32 &lt;==定义 to_localhost 可联机到本机
acl to_localhost dst ::1/128

# 2\. 信任用户与目标控制，定义可能使用这部 proxy 的外部用户(内网)
acl localnet src 10.0.0.0/8      &lt;==可发现底下都是 private IP 的设定
acl localnet src 172.16.0.0/12
acl localnet src 192.168.0.0/16
acl localnet src fc00::/7
acl localnet src fe80::/10
# 上述数据设定两个用户 (localhost, localnet) 与一个可取得目标 (to_localhost)

# 3\. 定义可取得的数据端口所在！
acl SSL_ports port 443                  &lt;==联机加密的埠口设定
acl Safe_ports port 80          # http  &lt;==公认标准的协议使用埠口
acl Safe_ports port 21          # ftp
acl Safe_ports port 443         # https
# 定义出 SSL_ports 及标准的常用埠口 Safe_ports 两个名称

# 4\. 定义这些名称是否可放行的标准依据(有顺序喔！)
http_access allow manager localhost  &lt;==放行管理本机的功能
http_access deny manager             &lt;==其他管理来源都予以拒绝
http_access deny !Safe_ports         &lt;==拒绝非正规的埠口联机要求
http_access deny CONNECT !SSL_ports  &lt;==拒绝非正规的加密埠口联机要求
&lt;==这个位置为你可以写入自己的规则的位置喔！不要写错了！有顺序之分的！
http_access allow localnet           &lt;==放行内部网络的用户来源
http_access allow localhost          &lt;==放行本机的使用
http_access deny all                 &lt;==全部都予以拒绝啦！

# 5\. 网络相关参数，最重要的是那个定义 Proxy 协议埠口的 http_port
http_port 3128     &lt;==Proxy 预设的监听客户端要求的埠口，是可以改的
# 其实，如果想让 proxy server/client 之间的联机加密，可以改用 https_port (923)

# 6\. 快取与内存相关参数的设定值，尤其注意内存的计算方式
hierarchy_stoplist cgi-bin ? &lt;==hierarchy_stoplist 后面的关键词 (此例为 cgi-bin)
# 若发现在客户端所需要的网址列，则不快取 (避免经常变动的数据库或程序讯息)
cache_mem 8 MB     &lt;==给proxy额外的内存，用来处理最热门的快取数据(需自己加)

# 7\. 磁盘高速缓存，亦即放置快取数据的目录所在与相关设定
cache_dir ufs /var/spool/squid 100 16 256 &lt;==默认使用 100MB 的容量放置快取
coredump_dir /var/spool/squid
# 底下的四个参数得要自己加上来喔！旧版才有这样的默认值！
minimum_object_size 0 KB    &lt;==小于多少 KB 的数据不要放快取，0 为不限制
maximum_object_size 4096 KB &lt;==与上头相反，大于 4 MB 的数据就不快取到磁盘
cache_swap_low 90   &lt;==与下一行有关，减低到剩下 90% 的磁盘高速缓存为止
cache_swap_high 95  &lt;==当磁盘使用量超过 95% 就开始删除磁盘中的旧快取

# 8\. 其他可能会用到的默认值！参考参考即可，并不会出现在配置文件中。
access_log /var/log/squid/access.log squid &lt;==曾经使用过 squid 的用户记录
ftp_user Squid@  &lt;==当以 Proxy 进行 FTP 代理匿名登录时，使用的账号名称
ftp_passive on   &lt;==若有代理 FTP 服务，使用被动式联机
refresh_pattern ^ftp:           1440    20%     10080
refresh_pattern ^gopher:        1440    0%      1440
refresh_pattern -i (/cgi-bin/&#124;\?) 0     0%      0
refresh_pattern .               0       20%     4320
# 上面这四行与快取的存在时间有关，底下内文会予以说明
cache_mgr root              &lt;==预设的 proxy 管理员的 email
cache_effective_user squid  &lt;==启动 squid PID 的拥有者
cache_effective_group squid &lt;==启动 squid PID 的群组
# visible_hostname &lt;==有时由于 DNS 的问题，找不到主机名会出错，就得加上此设定
ipcache_size 1024  &lt;==以下三个为指定 IP 进行快取的设定值
ipcache_low 90
ipcache_high 95
```

光是了解上述的一些基础设定值，可能就要头昏昏了，更别说 squid.conf 里面的其他设定值，看到头好昏... 无论如何，上述这些设定已经是很基础的设定了，你最好了解一下！除了 cache_dir 那一行取消批注，其他的保持不动！ 让我们以默认值来直接启动 squid 看看有什么特别的地方再说。

- 使用默认值来启动 squid 并观察相关信息

要启动 squid 真是简单，让我们来启动 squid 并且观察有没有相关的埠口吧！

```
[root@www ~]# /etc/init.d/squid start
init_cache_dir /var/spool/squid... 正在激活 squid: .       [  确定  ]
# 第一次启动会初始化快取目录，因此会出现上述左边的数据，未来这个讯息不会再出现
[root@www ~]# netstat -tulnp &#124; grep squid
Proto Recv-Q Send-Q Local Address   Foreign Address   State    PID/Program name
tcp        0      0 :::3128         :::*              LISTEN   2370/(squid)
udp        0      0 :::45470        :::*                       2370/(squid)
[root@www ~]# chkconfig squid on
```

如果你有设定 icp_port 时，squid 预设会启动 3128 及 3130 两个埠口，其中要注意的是， 实际帮用户进行监听与传送数据的是 port 3128 (TCP)，3130 (UDP) 仅是负责与邻近 Proxy 互相沟通彼此的快取数据库的功能，与实际的用户要求无关。因此，如果你的 proxy 是单纯的单一主机，或者是单纯的作为防火墙功能，那么这个 port 3130 是可以关闭的。就因如此，所以 CentOS 6.x 预设将这个设定值批注不使用啰！

例题：由于我的 Proxy 仅是部简单的单一代理服务器，并没有架设成为公开的邻近代理服务器 (peer proxy 或 neighbor proxy)， 因此想要关闭 port 3130 ，该如何处理？答：旧版的 CentOS 5.x 以前的版本才需要进行，很简单，直接修改 icp_port 即可！方法为：

```
[root@www ~]# vim /etc/squid/squid.conf
#Default: VBird 2011/04/06 modified，将下列数据从 3130 改为 0 即可
icp_port 0

[root@www ~]# /etc/init.d/squid restart
```

事实上，如果你的客户端与 proxy 之间的沟通想要使用加密机制的 SSL 功能，以保障客户端的信息避免被窃取时， 那么还有个 https_port 可以取代 http_port ！不过，充其量我们的 proxy 并非公开也仅是架设在内部区网， 因此还不需要使用到这个 https_port 啦！

- 观察与修改快取目录 (cache_dir)：权限与 SELinux

从前面的说明我们知道磁盘高速缓存是影响 proxy 效能的一个相当重要的参数，那么 squid 是如何将快取存进磁盘的呢？ squid 是将数据分成一小块一小块，然后分别放置到个别的目录中。由于较多的目录可以节省在同一个目录内找好多档案的时间 (想一想，分门别类的放置书籍在不同的书柜内，总比将所有书籍杂乱无章的放置到一个大书柜要好的多吧！)， 因此，在默认的 /var/spool/squid/ 目录下， squid 又会将它分成两层子目录来存放相关的快取数据，所以观察该目录就会是：

```
[root@www ~]# ls /var/spool/squid
00  01  02  03  04  05  06  07  08  09  0A  0B  0C  0D  0E  0F  swap.state
# 算一下，你会发现共有 16 个子目录！那么我们来看看第一个子目录的内容：

[root@www ~]# ls /var/spool/squid/00
00  08  10  18  20  28 ... 98  A0  A8  B0  B8  C0  C8  D0  D8  E0  E8  F0  F8
01  09  11  19  21  29 ... 99  A1  A9  B1  B9  C1  C9  D1  D9  E1  E9  F1  F9
....(中间省略)....
06  0E  16  1E  26  2E ... 9E  A6  AE  B6  BE  C6  CE  D6  DE  E6  EE  F6  FE
07  0F  17  1F  27  2F ... 9F  A7  AF  B7  BF  C7  CF  D7  DF  E7  EF  F7  FF
# 看见了吗？总共有 256 个子目录出现啰！
```

现在我们知道了较多的目录是为了将数据分门别类放置，但是第一层 16 个与第二层 256 个是怎么来的？ 让我们来瞧一瞧 cache_dir 这个重要参数的设定是怎样：

- cache_dir ufs /var/spool/squid 100 16 256

在 /var/spool/squid/ 后面的参数意义是：

- 第一个 100 代表的是磁盘使用量仅用掉该文件系统的 100MB
- 第二个 16 代表第一层次目录共有 16 个
- 第三个 256 代表每层次目录内部再分为 256 个次目录

根据 squid 的说法与其他文献的说明，这两层快取目录较佳的配置就是 16 256 以及 64 64 这两种配置， 所以我们也不需要修改相关的数据啦！重点时还得要注意这个目录的档案拥有者与 SELinux 类型才成呦！

例题：看起来预设的 proxy 的磁盘高速缓存应该是不够用，而之前的磁盘规划又没有做好，因此 /var/ 最多还有 500MB 可以让我们做为磁盘高速缓存。 那么如果想要将预设的磁盘高速缓存改为 500MB 而且再加上 /srv/squid/ 目录给予 2GB 的容量做为磁盘高速缓存，该如何进行设定？答：这里都与 cache_dir 有关！这个设定值可以重复出现多次！因此，我们可以这样进行的，特别注意底下的目录权限与 SELinux 类型呦！

```
[root@www ~]# vim /etc/squid/squid.conf
#Default: VBird 2011/04/06 modified，底下的设定除了拿掉 # 之外还得修改！
cache_dir ufs /var/spool/squid 500 16 256
cache_dir ufs /srv/squid 2000 16 256

[root@www ~]# mkdir /srv/squid
[root@www ~]# chmod 750 /srv/squid
[root@www ~]# chown squid:squid /srv/squid
[root@www ~]# chcon --reference /var/spool/squid /srv/squid
[root@www ~]# ll -Zd /srv/squid
drwxr-x---. squid squid system_u:object_r:squid_cache_t:s0 /srv/squid/

[root@www ~]# /etc/init.d/squid restart
```

之所以要改成 squid 拥有，是因为上头的 squid.conf 中，预设的启动 PID 的账号就是 squid 这个人物嘛！所以当然要变更！至于 SELinux 的类型方面，参考预设的 /var/spool/squid 就能够知道了。不过要注意，某些特定的目录 (例如 /home) 是不允许建立快取目录的， 因此我们使用服务资料可以放置的 /srv 作为测试范例啰！

想一想，既然快取是放在磁盘上面的，那么快取的数据会不会塞满整个快取磁盘呢？当然会啊！而且当塞满磁盘之后， 你的 proxy 恐怕就无法继续运作了！所以，我们当然得要好好的注意磁盘使用量是否已经饱和了。在上述的例题中， 若 /var/spool/squid 塞满 500MB 而 /srv/squid 塞满 2GB 那么你的 proxy 就挂了。为了避免这个问题，因此 squid 有底下两个重要设定：

- cache_swap_low 90
- cache_swap_high 95

代表当磁盘使用量达 95% 时，比较旧的快取数据将会被删除，当删除到剩下磁盘使用量达 90% 时，就停止持续删除的动作。 以本案例中，总共 2.5GB 的容量，当用到 2.5*0.95=2.375G 时，旧的数据会开始被删除，删到剩下 2.5*0.9=2.25GB 时，就停止删除的意思。所以会被删除掉 125MB 的旧数据就是了。通常这个设定值已经足够了，不需要变动他，除了你的快取太大或太小时，才会调整这个设定值。

- squid 使用的内存计算方式

事实上，除了磁盘容量之外，内存可能是另一个相当重要的影响 proxy 效能的因子！怎么说呢？因为 proxy 会将数据存一份在磁盘高速缓存中，但是同时也会将数据暂存在内存当中啊，以加快未来使用者存取同一份数据的速度！ 但是这个内存快取是需要花费额外的服务器物理内存的量，所以就得要以额外的设定值来指定啰。那就是 cache_mem 这个设定值的功能了。

很多人 (包括鸟哥) 都会误会 cache_mem 的用途！其实 cache_mem 是额外的指定一些内存来进行比较『热门』的数据存取！ cache_mem 并不是指我要使用多少内存给 squid 使用，而是指 "我还要额外提供多少内存给 squid 使用" 的意思』！由于预设 1GB 的磁盘高速缓存会占用约 10M 的内存，而 squid 本身也会占用约 15MB 的内存， 因此，上个例题中 squid 使用掉的内存就有：

- 2.5 * 10 + 15 + "cache_mem 设定值 (8)"

squid 官方网站建议你的物理内存最好是上面数值的两倍，也就是说，上述的内存使用量已经是 48MB， 则我的物理内存最好至少要有 100 MB 以上，才会有比较好的效能！当然，这个单指 Proxy 部分而已，如果你的该部主机还有负责其他的工作，呵呵！那么内存就得在累加上去啦！一般来说，如果你的 Proxy 很多人使用时，这个值越大越好，但是最好也要符合上面的需求喔！

例题：由于我的内存够大，而 proxy 确实是我重要的服务，因此想要增加额外的 32MB 作为热门数据快取，该如何修改？答：直接做了啦！就是修改 cache_mem 而已！

```
[root@www ~]# vim /etc/squid/squid.conf
#Default: VBird 2011/04/06 modified，将原本的 8 改为 32 啰！
cache_mem 32 MB

[root@www ~]# /etc/init.d/squid restart
```

17.2.3 管控信任来源 (如区网) 与目标 (如恶意网站)： acl 与 http_access 的使用

在上面的基础设定中，其实仅有 proxy 服务器本身可以向自己的 proxy 要求网页代理～那有个屁用啊？ 我们的重点是想要开放给区网来使用这个 proxy 的嘛！所以当然得要修改信任用户的管控参数啰。 此时，那个重要到不行的 acl 就得要来瞧一瞧啦！这个 acl 的基本语法为：

```
acl &lt;自定义的 acl 名称&gt; &lt;要控制的 acl 类型&gt; &lt;设定的内容&gt;
```

由于 squid 并不会直接使用 IP 或网域来管控信任目标，而是透过 acl 名称来管理，这个 <acl 名称> 就必须要设定管理的是来源还是目标 (acl 类型) ，以及实际的 IP 或网域 (设定的内容) 啦！这个 acl 名称可以想成是一个昵称就是了。那么有哪些重要的 acl 类型呢？基本上有这些：

- 管理是否能使用 proxy 的信任客户端方式：

由于因特网主要有使用 IP 或主机名来作为联机方式的，因此信任用户的来源至少就有底下几种：

- src ip-address/netmask： 主要控制『来源的 IP 地址』。举例来说，鸟哥的内网有两个，分别是 192.168.1.0/24 以及 192.168.100.0/24 ， 那么假设我想要制订一个 vbirdlan 的 acl 名称，那就可以在配置文件内写成： acl vbirdlan src 192.168.1.0/24 192.168.100.0/24

- src addr1-addr2/netmask： 主要控制『一段范围来源的 IP 地址』。假设我只想要让 192.168.1.100-192.168.1.200 使用这部 proxy ，那么就用： acl vbirdlan2 src 192.168.1.100-192.168.1.200/24

- srcdomain .domain.name： 如果来源用户的 IP 一直变，所以使用的是 DDNS 的方式来更新主机名与 IP 的对应，此时我们可以使用主机名来开放！ 例如来源是 .ksu.edu.tw 的来源用户就开放使用权，那就是： acl vbirdksu srcdomain .ksu.edu.tw

- 管理是否让 proxy 帮忙代理到该目标去获取数据：

除了管理来源用户之外，我们还能够管理是否让 proxy 服务器到某些目标去获取数据喔！在预设的设定中， 我们的 proxy 仅管理可以向外取得 port 21, 80, 440... 等端口的目标网站，不是这些端口就无法帮忙代理取得。 至于 IP 或网域则没有管理。基本的管理有这些方式：

- dst ip-addr/netmask： 控制不能去的目标网站的 IP ，举例来说，我们不许 proxy 去捉取 120.114.150.21 这部主机的 IP 时，可以写成是： acl dropip dst 120.114.150.21/32

- dstdomain .domain.name： 控制不能去的目标网站的主机名。举例来说，如果你在上课时不允许学生跑去种田还是小小战争，那就得要把 .facebook.com 给关闭！那就需要写成： acl dropfb dstdomain .facebook.com
- url_regex [-i] ^http://url： 使用正规表示法来处理网址列的一种方式！这种方式的网址列必须要完整的输入正规表示法的开始到结尾才行。 举例来说，昆山科大的中文网页写法为 (并非部分比对，所以最结尾的 . 记得要加上去！)： *acl ksuurl url_regex ^http://www.ksu.edu.tw/cht/.*
- urlpath_regex [-i] .gif$： 与上一个 acl 非常类似，只是上一个需要填写完整的网址数据，这里则是根据网址列的部分比对来处置。以上述的预设案例来说， 只要网址列结尾是 gif (图片文件) 就符合这个项目了。万一我要找出有问题的色情网站，有出现 /sexy 名称并以 jpg 结尾的， 就予以抵挡，那就是使用： acl sexurl urlpath_regex /sexy.*.jpg$

除了上述的功能之外，我们还能够使用外部的档案来提供相对应的 acl 内容设定值喔！ 举例来说，假设我们想要抵挡的外部主机名常常会变动，那么我们可以使用 /etc/squid/dropdomain.txt 来设定主机名， 然后透过底下的方式来处理

```
acl dropdomain dstdomain "/etc/squid/dropdomain.txt"
```

然后在 dropdomain.txt 当中，一行一个待管理的主机名，这样也能够减少持续修改 squid.conf 的困扰！ 好了！了解了 acl 之后，接下来得要谈谈 http_access 这个实际放行或拒绝的参数了！

- 以 http_access 调整管理信任来源与管控目标的『顺序』：

设定好 acl 之后，接下来就是要看看到底要不要放行喔～放行与否跟 http_access 这个项目有关。基本上， http_access 就是拒绝 (deny) 与允许 (allow) 两个控件目，然后再加上 acl 名称就能够达到这样的功能了！ 只是你得要特别注意的是：http_access 后面接的数据，是有顺序的！这个观念很重要喔！ 我们用底下的案例来说明好了：

假设我要放行内部网络 192.168.1.0/24, 192.168.100.0/24 这两段网域，然后拒绝对外的色情相关图片， 以及 facebook.com 网站，那么就应该要这样做：

```
[root@www ~]# vim /etc/squid/squid.conf
# http_access 是有顺序的，因此建议你找到底下这个关键词行后，将你的资料加在后面
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
acl vbirdlan src 192.168.1.0/24 192.168.100.0/24
acl dropdomain dstdomain .facebook.com
acl dropsex urlpath_regex /sexy.*jpg$
http_access deny dropdomain  &lt;==这三行的『顺序』很重要！
http_access deny dropsex
http_access allow vbirdlan

[root@www ~]# /etc/init.d/squid restart
```

你得要注意，如果先放行了 vbirdlan 才抵挡 dropdomain 时，你的设定可能会失败！因为内网已经先放行，因此后面的规则不会比对，那么 facebook.com 就无法被抵挡了！这点得要很注意才行！通常的作法是，先将要拒绝的写上去，然后才写要放行的数据就好了。

17.2.4 其他额外的功能项目

- 不要进行某些网页的快取动作

从前面的说明我们知道 Proxy 的快取通常在记录比较少变动的数据，如果是讨论区或者是程控类的数据库型态网页， 那么恐怕就没有快取的需要，因为数据一直变动嘛！你总不希望你发了一帖留言，结果等一下再去浏览时，看到的还是旧留言吧！ 所以啰，在预设的情况下，squid 已经拒绝某些数据的快取了，那就是底下的几个设定值：

```
acl QUERY urlpath_regex cgi-bin \?
cache deny QUERY  &lt;==重点就是这一行！可以拒绝，不要让后面的 URL 被快取！
```

我们知道通常 .php 结尾的网页大部分就是讨论区之类的变动性数据，那么能不能出现 .php 结尾的网页就不要快取呢？ 当然可以啊！那该如何进行？我们以上面的数据来照样造句一下吧！

例题：只要网址列出现 .php 结尾的，就不予以快取！答：透过 acl 配合 cache 这两个参数来处理即可！

```
[root@www ~]# vim /etc/squid/squid.conf
acl denyphp urlpath_regex \.php$
cache deny denyphp
# 在此档案的最后新增这两行即可！

[root@www ~]# /etc/init.d/squid restart
```

- 磁盘中快取的存在时间

还记得底下的设定值吗？这个设定值的参数是这样设定的：

```
# refresh_pattern &lt;regex&gt;   &lt;最小时间&gt; &lt;百分比&gt; &lt;最大时间&gt;
refresh_pattern ^ftp:           1440    20%     10080
refresh_pattern ^gopher:        1440    0%      1440
refresh_pattern -i (/cgi-bin/&#124;\?) 0     0%      0
refresh_pattern .               0       20%     4320
```

- regex：使用的是正规表示法来分析网址列的资料，如上面第一行设定为网址列开头是 ftp 的意思。
- 最小时间：单位是分钟，当取得这个数据的时间超过这个设定值，则该数据会被判定为旧资料。如上面第一行， 表示当取得的资料超过 1440 分钟时，该资料会被判定为旧数据，若有人尝试读取同样的网址列，那么 squid 会重新抓取该数据，不会使用快取内的

旧数据。至于第三行，则表示除了上述的两个开头数据外，其他的数据都是被定义为新的， 因此 squid 只会从快取内抓数据给客户端。

- 百分比：这个项目与『最大时间』有关，当该资料被抓取到快取后，经过最大时间的多少百分比时，该数据就会被重抓。
- 最大时间：与上一个设定有关，就是这个数据存在快取内的最长时间。如上面第一行，最大时间为 10080 分钟，但是当超过此时间的 20% (2016分钟) 时，这个数据也会被判定为旧资料。

例题：在网址列出现 .vbird. 字样时，该数据为暂时使用的，因此 2 小时后就算旧数据。而最长保留在快取给她一天的时间， 且经过 50% 的时间后，就被判定为旧数据吧！答：

```
[root@www ~]# vim /etc/squid/squid.conf
refresh_pattern ^ftp:           1440    20%     10080
refresh_pattern ^gopher:        1440    0%      1440
refresh_pattern -i (/cgi-bin/&#124;\?) 0     0%      0
refresh_pattern \.vbird\.       120     50%     1440
refresh_pattern .               0       20%     4320

[root@www ~]# /etc/init.d/squid restart
```

- 主机名与管理员的 email 指定

如果你的服务器主机名尚未决定，因此使用的主机名在因特网上面是找不到对应的 IP 的 (因为 DNS 未设定)， 那么在预设的 squid 设定中，恐怕会无法顺利的启动。此时你可以手动的加入一个主机名，就是透过 visible_hostname 来指定。 同时，如果客户端使用 squid 出现任何错误时，屏幕上都会出现管理员的 email 让用户可以回报。现在假设主机名为 www.centos.vbird 且管理员的 email 为 dmtsai@www.centos.vbird ，此时我们可以这样修改：

```
[root@www ~]# vim /etc/squid/squid.conf
cache_mgr dmtsai@www.centos.vbird  &lt;==管理员的 email 呦！
visible_hostname www.centos.vbird  &lt;==直接设定主机名喔！

[root@www ~]# /etc/init.d/squid restart
```

17.2.5 安全性设定：防火墙, SELinux 与黑名单档案

- 防火墙得要放行 tcp 的 port 3128

现在我们已经设定了让 192.168.100.0/24 及 192.168.1.0/24 这两段来源使用我们的 proxy server ， 那么想当然尔，防火墙的设定就得要开放这两段使用 port 3128 才行啊！不过你得要特别注意，并不是开放防火墙就能使用 proxy server 的资源，还得要使用 acl 配合 http_access 才行呦！注意注意！假设你已经使用了 iptables.rule ， 那么修改的方法就是这样：

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.allow
iptables -A INPUT -i $EXTIF -p tcp -s 192.168.1.0/24 --dport 3128 -j ACCEPT
# 因为内网 192.168.100.0/24 本来就是全部都接受放行的！

[root@www ~]# /usr/local/virus/iptables/iptables.rule
```

- SELinux 的注意事项

针对 proxy 来说，CentOS 6.x 倒是没有给予太多的规则限制，因此似乎不太需要修订规则。不过，SELinux 的安全本文在类型部分得注意。这包括配置文件 (/etc/squid/ 内的数据) 类型是 squid_conf_t 的样式， 而快取目录的类型则是 squid_cache_t 的类型，且上层类型 (/var/spool/) 应该是要成为 var_t 之类的才行。 修改的方法就是透过 chcon 来处理即可。

- 建立黑名单配置文件

我们在 17.2.3 小节里面谈到，可以透过『 dstdomain .domain.name 』来抵挡不想联机的网站。 不过每次都得使用 root 身份来设定 squid.conf 才行。那有没有办法额外处理出一个档案，让想要拒绝联机的数据写入， 这样比较容易管理，不需要一直去修改 squid.conf 嘛！有没有办法可以达成呢？有的，就透过特定档案来处置即可。 看看底下这个例题来修订一下吧：

例题：建立一个名为 /etc/squid/dropdomain.txt 的档案，内容为拒绝联机的目标网站。答：我们之前设定过相关的网站，处理的方法是直接将主机名写入 squid.conf 中，现在我们可以这样修订：

```
[root@www ~]# vim /etc/squid/squid.conf
# 找到底下的数据，就是 dropdomain 那行，约在 629 行左右，并且修改一下
acl dropdomain dstdomain "/etc/squid/dropdomain.txt"
# 注意一下，如果是档名，请写绝对路径，且使用双引号或单引号圈起来！

[root@www ~]# vim /etc/squid/dropdomain.txt
.facebook.com
.yahoo.com
# 一行一个 domain 名称即可

[root@www ~]# /etc/init.d/squid reload
```

这个方法的好处是，你可以使用额外的控制方式去修改 /etc/squid/dropdomain.txt 这个档案的内容， 并且修改完毕后再使用 reload 去加载配置文件，不必要重新启动 (restart)，因为 reload 的速度比较快速。 举例来说，鸟哥的专题生就用 PHP 写了一支控制该档案的网页接口，可以让老师在上课时直接透过网页输入要被控制的目标网站， 这样学生就无法在上课时联机到外面的某些网站去玩游戏啰～

17.3 客户端的使用与测试

既然 proxy 是给浏览器用的，那么自然在浏览器上面就需要设定一些参数啰！那么如何设定呢？由于不同的浏览器在设定 Proxy 的地方也都不同，所以底下我们介绍目前比较常见的两款浏览器，分别是 firefox 以及 IE 的设定，至于其他的浏览器，请参考各浏览器的相关说明啊！

17.3.1 浏览器的设定： firefox & IE

- firefox 5.x 的设定示意

要在 firefox 5.X 上面设定好 proxy 基本步骤是这样的：首先打开 firefox 软件，出现如下的图标后，点选：『工具』内的『选项』， 示意画面如下所示：

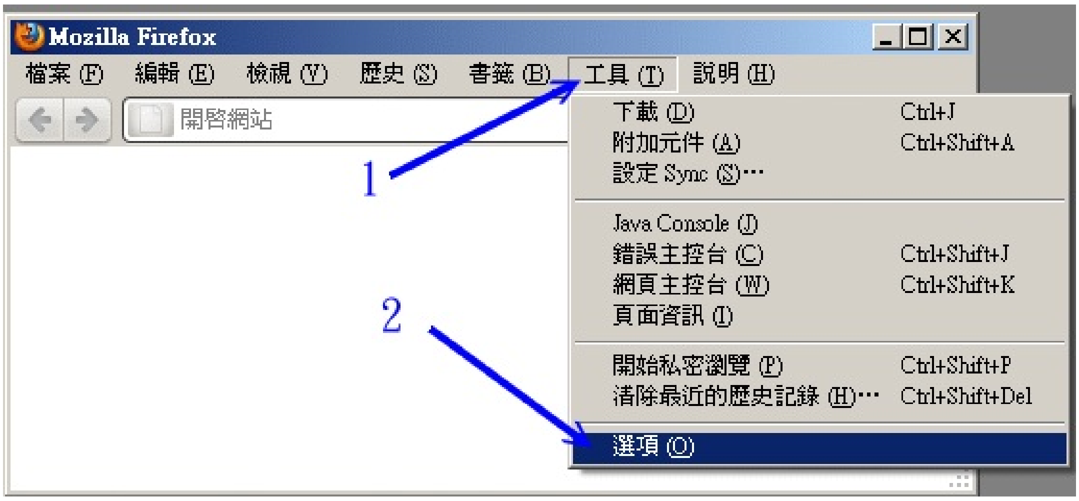


图 17.3-1、在 firefox 上头设定 proxy 的流程

然后在出现的如下画面中，先选择右上方的『进阶』项目，然后点选『网络』页面，最后再点选联机的『设定』按钮， 如下图所示，依序来动作：

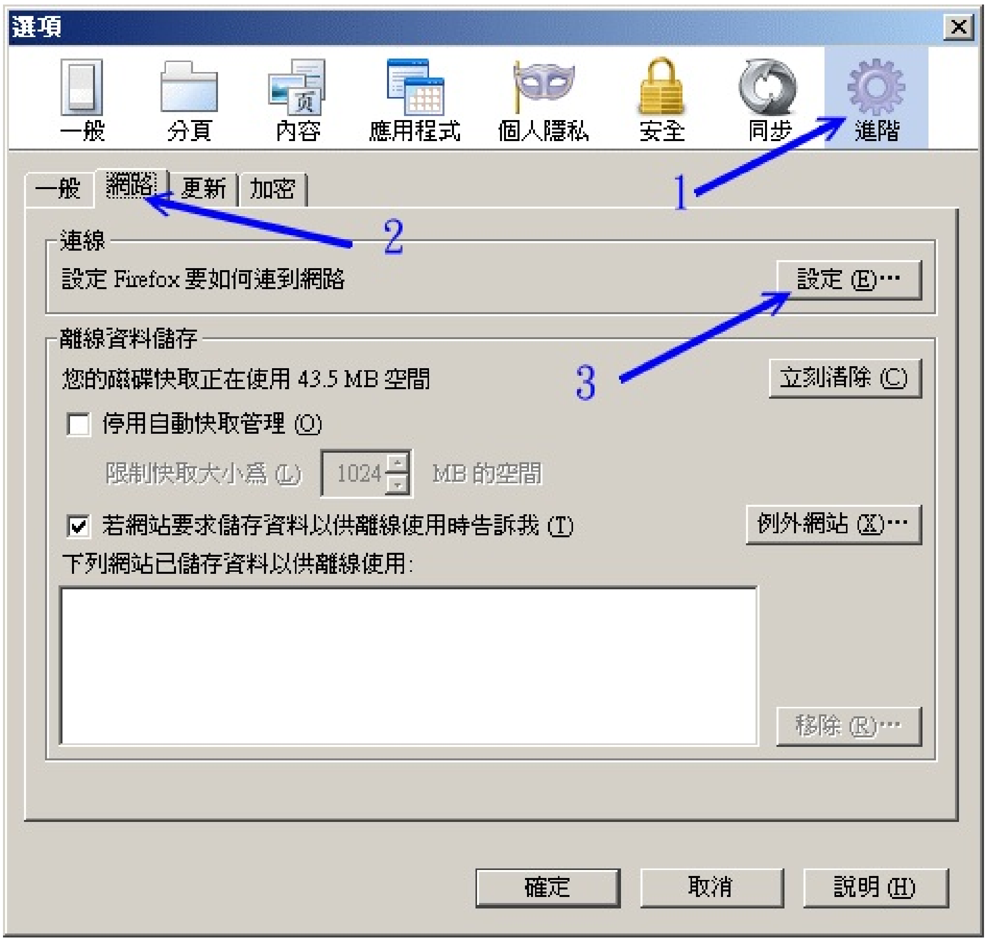


图 17.3-2、在 firefox 上头设定 proxy 的流程

此时就会出现如下图所示的要你输入代理服务器的相关数据。请先点选『手动设定』之后才能够填写底下的方格。 填上我们服务器的 IP (鸟哥的案例中，使用的是 192.168.1.100 这一部) 以及埠口，然后鸟哥建议你也可以勾选『所有通讯协议都用此 proxy 』的项目，都设定妥当后，才按下确定。如下图所示的流程：

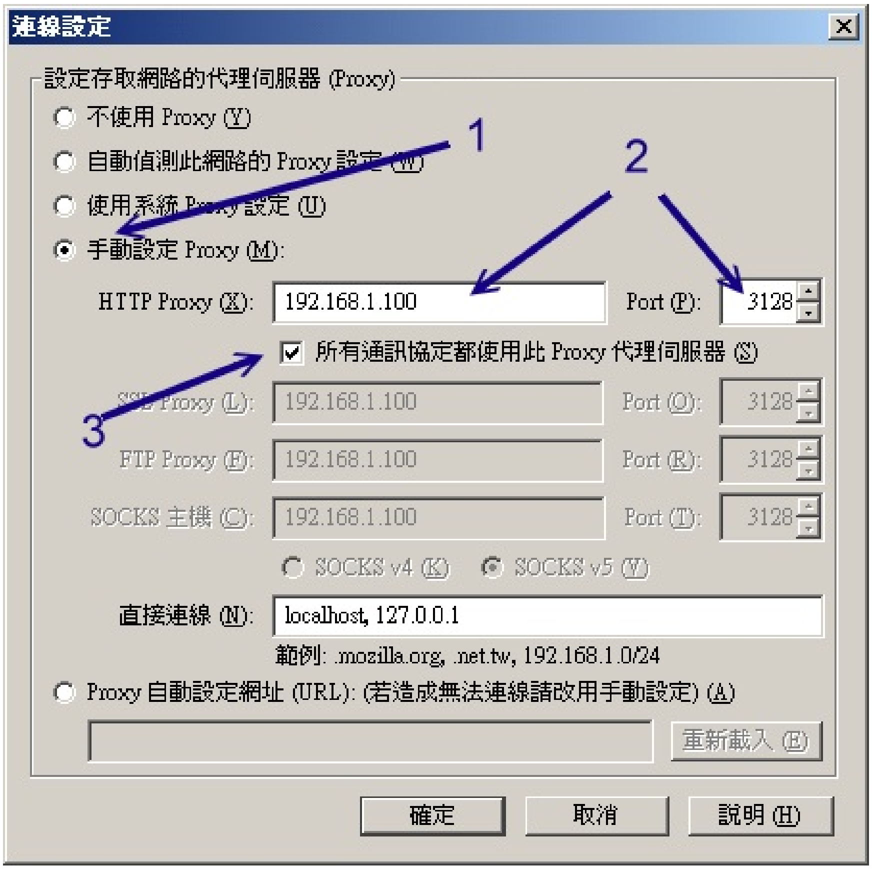


图 17.3-3、在 firefox 上头设定 proxy 的流程

这样就设定好 firefox 的 proxy 相关数据了，有够简单吧！

- IE 的设定示意

那么 IE 要怎么设定呢？也是很简单啦！首先，打开 IE 软件，你会看到如下的示意图，点选『工具』内的『因特网选项』， 流程如下所示：

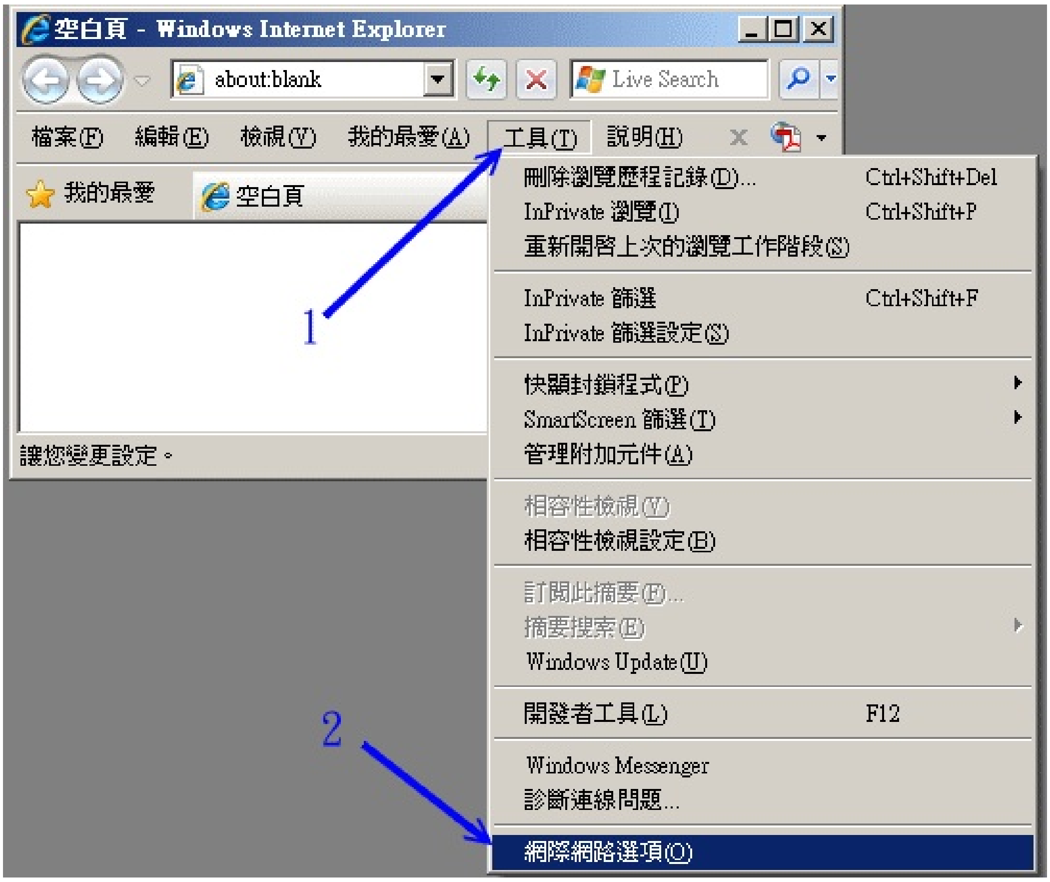


图 17.3-4、在 IE 上头设定 proxy 的流程

在接下来的窗口中，点选『联机』的页面，然后按下『局域网络设定』的按钮。流程如下所示：

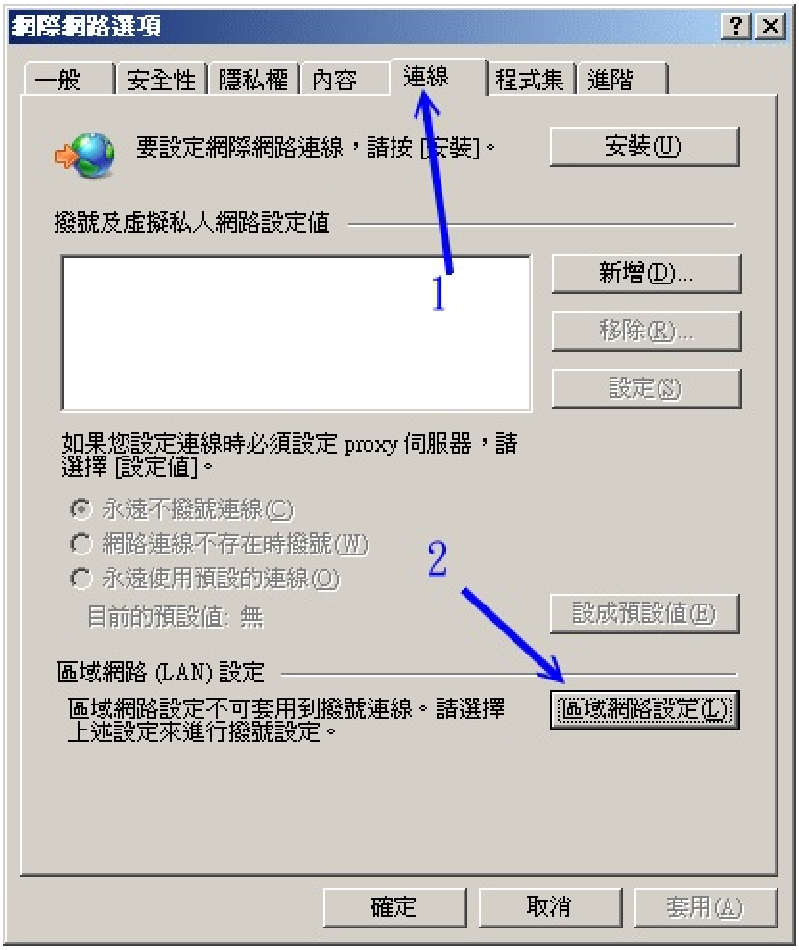


图 17.3-5、在 IE 上头设定 proxy 的流程

最后就是要输入正确的 proxy server 的 IP 与 port 的相关数据啊！如下图所示，先点选箭头 1 所指定的项目，然后才能够开始填写正确数据。 一般来说，近端网址 (例如区网的服务器) 可以不透过 proxy 去捉取数据，因此这里可以勾选箭头三所示意的方框喔！ 这样就设定完毕。

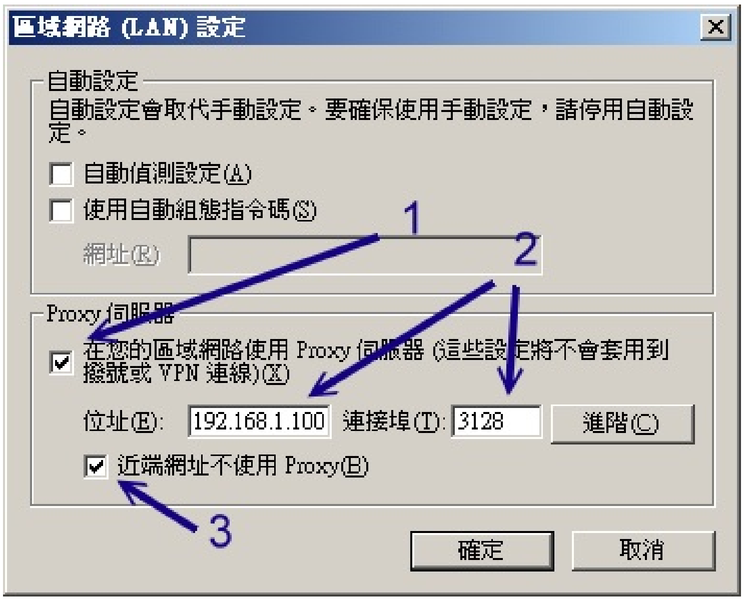


图 17.3-6、在 IE 上头设定 proxy 的流程

接下来让鸟哥用 firefox 来测试一下，如果你要连的网站是被拒绝的会如何？

17.3.2 测试 proxy 失败的画面

开始利用你的浏览器来浏览各个网站，基本上你都会发现正确的网站内容。但如果你要连的网站是刚刚被拒绝的呢？ 举例来说，刚刚我们有设定拒绝连向 .yahoo.com 的喔！那么如果你真的输入网址是 tw.yahoo.com，那屏幕上应该是会这样输出的！



图 17.3-7、联机被 proxy 拒绝时的反应情况

从上图我们可以发现，目标网站是 tw.yahoo.com，然后产生问题的地方在于『 存取被拒绝 (Access Denied) 』，表示问题的发生在于 proxy 的设定，然后系统还很好心的告诉你管理员 (cache administrator) 的 email ，让你有问题可以回报给他。 最后，这个信息是否为新的？底下还会告诉你这个错误发生的时间点呢！这样有没有很清楚啊？ ^_^！ proxy 的错误不只是这些，因此，当你还有发现无法联机的网站时，请务必要看看屏幕的输出信息才好呦！

17.4 服务器的其他应用设定

除了基本的 proxy 设定之外，如果你还有其他可供利用的上层代理服务器，说不定我们就能够设计一下如何进行分流的动作了！ 此外，如果针对信任用户来说，难道得要一直使用 acl 直接指定用户来源然后再用 http_access 放行？有没有认证功能啊？ 这样就不用一直修改设定啊！这些其它的应用设定在这个小节来谈谈吧！

17.4.1 上层 Proxy 与获取数据分流的设定

能够找到的上层 proxy 服务器我们在 17.1.3 里面谈过了，你可以重新回去瞧瞧。 不过，假设你所在的环境并没有上层代理服务器，但是你有两部 Linux 主机放置在不同的 ISP 环境下，这两个 ISP 对某些国外的带宽流量不同，所以你想要根据这样的情况来设计一下获取 WWW 网页的分流时，可以怎么做？ 我们举个例子来说好了：

- hinet.centos.vbird：这部主机位于 hinet 这个 ISP 底下，对大陆 (.cn) 的流量比较高，作为上层代理服务器之用；
- www.centos.vbird：这部主机位于学术网络 (昆山科大)，因为对大陆带宽被限制，因此浏览速度相对较慢。

现在我们规划 hinet.centos.vbird 是上层代理服务器，因此这部主机得要开放 www.centos.vbird 这部机器的使用权， 这动作包括： (1)利用 acl srcdomian 等方式放行 www.centos.vbird 的使用权； (2)开放 www.centos.vbird 的 port 3128 的防火墙过滤功能。如此一来，我们这部 www.centos.vbird 才能够使用上层代理服务器喔！也就是说，这两部主机都要是你能够掌握的才行 (至少也要上层 ISP 能够替你开放使用权啦)。

那么 www.centos.vbird 要如何设定呢？基本上，设定上层代理服务器与分流的参数主要有：cache_peer, cache_peer_domain, cache_peer_access 等，分别说明语法如下：

- cache_peer 的相关语法

```
cache_peer [上层proxy主机名] [proxy角色] [proxy port] [icp port] [额外参数]
```

这个设定值就是在规范上层代理服务器在哪里，以及我们想要对这部代理服务器如何查询的相关设定值。

- 上层 proxy 主机名：例如本案例中就是 hinet.centos.vbird 这一部啰；
- proxy 角色：这部 proxy 是我们的上层 (parent) ？还是作为我们邻近 (sibling) 的协力运作的 proxy ？ 因为我们要利用上层去捉取数据，因此经常使用的是 parent 这个角色值；
- proxy port：通常就是 3128 嘛！
- icp port：通常就是 3130 嘛！

- 额外参数：针对这部上层 proxy 我们想要对它进行的查询数据的行为设定。主要有：
  - proxy-only：向上层 proxy 要到的数据不会快取到本地的 proxy 服务器内，降低本地 proxy 负担；
  - wieght=n：权重的意思，因为我们可以指定多部上层 Proxy 主机，哪一部最重要？就可以利用这个 weight 来设定，n 越大表示这部 Proxy 越重要
  - no-query：如果向上层 Proxy 要求资料时，可以不需要发送 icp 封包，以降低主机的负担
  - no-digest：表示不向附近主机要求建立 digest 纪录表格
  - no-netdb-exchange：表示不向附近的 Proxy 主机送出 imcp 的封包要求
- cache_peer_domain 的相关语法

```
cache_peer_domain [上层proxy主机名] [要求的领域名]
```

这个设定值的意思是说，你想要使用这部上层代理服务器向哪个领域名要求数据。

- cache_peer_access 的相关语法

```
cache_peer_access [上层proxy主机名] [allow&#124;deny] [acl名称]
```

与 cache_peer_domain 相当类似，只是 cache_peer_domain 直接规范了主机名 (domain name)， 而如果你想要设计的并非领域名，而是某些特定的 IP 网段时，就得要先用 acl 设计一个名称后， 再以这个 cache_peer_access 去放行 (allow) 或拒绝 (deny) 读取了。

根据上述的语法说明，那么我们想要达到 .cn 使用 hinet.centos.vbird 这部服务器的代理功能时， 应该要这样设计的：

```
[root@www ~]# vim /etc/squid/squid.conf
cache_peer hinet.centos.vbird parent 3128 3130 proxy-only no-query no-digest
cache_peer_domain hinet.centos.vbird .cn

[root@www ~]# /etc/init.d/squid reload
```

如果你还有其它的需求再利用 acl 规范了目标位置后，再以 cache_peer_access 去放行吧！如此一来，你的 proxy server 就是一部会主动的依据不同的要求向不同的上层服务器求取数据的聪明 proxy 啰！

17.4.2 Proxy 服务放在 NAT 服务器上：通透式代理 (Transparent Proxy)

从上面的说法来看，我们可以发现 proxy 可以做到类似防火墙的功能 (acl dst, acl dstdomain 再配合 http_access 处理)， 但是，我们也知道浏览器得要设定好 proxy 之后，才会真的使用 proxy 嘛！那就不就是在耍宝用的防火墙吗？ 只要你的用户知道不要设定 proxy 就可以躲过你的管控，那这部 proxy 防火墙有啥屁用啊？您说是吧？

那该如何强制使用者一定要使用你的 proxy 呢？很简单！那就是： (1)在对外的防火墙服务器 (NAT) 上面安装 proxy； (2)在 proxy 上头启动 transparent 功能； (3) NAT 服务器加上一条 port 80 转 port 3128 的规则，如此一来，所有往 port 80 的封包就会被你的 NAT 转向 port 3128 ， 而你的 port 3128 就是 proxy ，那大家就得要用你的 proxy ，而且重点是，浏览器不需要进行任何设定！

呵呵！也就是说，当使用者是经过 NAT 服务器联机出去时，只要让 NAT 服务器发现『咦！你是要去捉 WWW 的资料对吧！好！那么这个动作由 Proxy 服务帮你搞定！』如此一来，使用者根本就不需要在浏览器上面设定 Proxy 的相关数据，因为这个动作是『由 NAT 服务器自己决定的』，所以只要在 NAT 服务器上面设定妥当即可，使用者不必设定任何数据呢！呵呵！真是不错！而且进行的动作非常简单！

```
# 1\. 设定 proxy 成为通透式代理服务器的功能！
[root@www ~]# vim /etc/squid/squid.conf
http_port 3128  transparent
# 找到 3128 这行后，在最后面加上 transparent 即可

[root@www ~]# /etc/init.d/squid reload
```

接下来，将来自 192.168.100.0/24 这个内网的来源，只要是要求 port 80 的，就将它重新导向 port 3128 的方式为：

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.rule
iptables -t nat -A PREROUTING -i $INIF -s 192.168.100.0/24 -p tcp \
         --dport 80 -j REDIRECT --to-ports 3128
# 将上述这一行加在最底下 /etc/init.d/iptables save 的上面一行即可！

[root@www ~]# /usr/local/virus/iptables/iptables.rule
```

这样就结束啦！很简单吧！通常这样的环境相当适合学校内的教室或者是计中的环境， 因为这样学校内部根本不需要请学生设定浏览器的 proxy 功能，立刻就能够达到我们所需要的管控能力！很棒吧！ 不过，虽然这样的功能已经很棒了，但是鸟哥实际用在学校教室环境中却发现了一些问题， 那就是很多同学同时上传同一个档案到外部服务器去，因为 proxy 快取的功能，结果让学生一直取得旧的档案， 对于教网页制作的老师来说，很困扰～因为教学过程中常常需要上传最新的网页嘛！但是 proxy 快取住， 所以却得到错误的数据了～那怎办？

- 仅具有 proxy 无快取功能的代理

既然我们这个 transparent proxy 的目的仅是在进行控管，并不要去处理快取的任务 (因为带宽假设是够的)， 那么干脆就不要快取啦！这样不就 OK 啦？好吧！那我们就来搭配 transparent 进行这个设定看看。 假设 transparent proxy 已经设定妥当，那么接下来就是让

你的快取目录空空如也，且再也不写入任何资料。 此外，也不要有多余的内存来记录热门档案啦！

```
# 先关闭 squid ，然后删除快取目录，之后再重建快取目录，此时快取目录就空了
[root@www ~]# /etc/init.d/squid stop
[root@www ~]# rm -rf /var/spool/squid/*
[root@www ~]# vim /etc/squid/squid.conf
cache_dir ufs /var/spool/squid 100 16 256 &lt;u&gt;read-only&lt;/u&gt;
#cache_dir ufs /srv/squid 2000 16 256
# 额外的那个 /srv/squid 批注掉，然后第一行多个 read-only 字样！
cache_mem 0 MB
# 本来规范有 32MB ，现在不要了！

[root@www ~]# /etc/init.d/squid start
```

如此一来，这部 proxy 就再也没有快取了，全部资料都得要自己向外头捉取！就不会有旧数据重复出现的问题～

17.4.3 Proxy 的认证设定

既然 proxy 有许多功用，包括分流的功能，很不赖啊！但是，由于网络闲人越来越多，因此 proxy 不可以设计为 open proxy ！亦即是不能够开放所有的人使用你的 proxy 啦！所以，一般来说， proxy 只会开放内部网域的人们来使用而已。 问题是，如果我在 Internet 也想要使用这部自己架设的 proxy 时，该如何是好？还得要再次的修改 squid.conf 吗？ 有没有这么麻烦？

没关系啦！为了这个问题， squid 官方软件已经给予了认证的设定功能！意即我们可以透过认证来简单的输入账号密码， 若通过验证，就可以立刻使用我们的 proxy 了！这样就好多啦！那如何达成呢？其实 squid 提供很多认证功能， 我们需要的是最简单的功能即可。使用的是 squid 主动提供的 ncsa_auth 认证模块，这个模块会利用 apache (WWW 服务器) 提供的帐密建立指令 (htpasswd) 所制作的密码文件作为验证依据。所以，我们至少需要检查有没有这两样东西：

```
[root@www ~]# rpm -ql squid &#124; grep ncsa
/usr/lib64/squid/ncsa_auth    &lt;==有的！就是这个验证模块档案！注意完整路径
/usr/share/man/man8/ncsa_auth.8.gz

[root@www ~]# yum install httpd   &lt;==apache 软件安装
[root@www ~]# rpm -ql httpd &#124; grep htpasswd
/usr/bin/htpasswd           &lt;==就是需要这个帐密建立指令！
/usr/share/man/man1/htpasswd.1.gz
```

这样的事前准备就差不多了。让我们来考虑一个案例好了：

- 内部网域 192.168.100.0/24 要使用 proxy 的，还是不需要透过验证；
- 外部主机想要使用 proxy (例如 192.168.1.0/24 这段) 才需要验证；
- 使用 NCSA 的基本身份验证方式，且密码文件建立在 /etc/squid/squid_user.txt
- 上述档案仅有一个用户 vbird ，他的密码为 1234

那该如何处理呢？开始来一步一步进行吧：

```
# 1\. 先修改 squid.conf 档案内容
[root@www ~]# vim /etc/squid/squid.conf
# 1.1 先设定验证相关的参数
auth_param basic program /usr/lib64/squid/ncsa_auth /etc/squid/squid_user.txt
auth_param basic children 5
auth_param basic realm Welcome to VBird's proxy-only web server
# 非特殊字体为关键词不可更动，第一行为透过 ncsa_auth 读取 squid_user.txt 密码
# 第二行为启动 5 个程序 (squid 的子程序) 来管理验证的需求；
# 第三行为验证时，显示给用户看的欢迎讯息，这三行可写在最上面！

# 1.2 然后是针对验证功能放行与否的 acl 与 http_access 设定
acl vbirdlan src 192.168.100.0/24  &lt;==修改一下，取消 192.168.1.0/24
acl dropdomain dstdomain "/etc/squid/dropdomain.txt"
acl dropsex urlpath_regex /sexy.*jpg$
acl squid_user proxy_auth REQUIRED &lt;==建立一个需验证的 acl 名称
http_access deny dropdomain
http_access deny dropsex
http_access allow vbirdlan
http_access allow squid_user       &lt;==请注意这样的规则顺序喔！验证在最后

# 2\. 建立密码数据
[root@www ~]# htpasswd -c /etc/squid/squid_user.txt vbird
New password:
Re-type new password:
Adding password for user vbird
# 第一次建立才需要加上 -c 的参数，否则不需要加上 -c 喔！

[root@www ~]# cat /etc/squid/squid_user.txt
vbird:vRC9ie/4E21c.  &lt;==这就是用户与密码啰！

[root@www ~]# /etc/init.d/squid restart
```

比较需要注意『acl squid_user proxy_auth REQUIRED』这一串设定，proxy_auth 是关键词，而 REQUIRED 则是指定任何在密码文件内的用户都能够使用验证的意思。如果一切顺利的话，那么你的内网依旧可以使用 transparent proxy ，而外网则需要输入账密才能够使用 proxy server 提供的代理能力。至于验证的过程有点像这样：



图 17.4-1、使用 proxy 需验证的示意图

上图中箭头 1 为刚刚你设定的 real 内容，而帐密则是你用 htpasswd 所建立的数据啦！另外，既然已经加上了验证功能，那么你可能得要将防火墙开放 port 3128 对全世界监听的过滤才行吻！防火墙还是不要忘记了！ ^_^

17.4.4 末端登录档分析： sarg

事实上， squid 已经收集了众多的登录文件分析软件了，而且大多是免费的 (http://www.squid-cache.org/Scripts/) ，你可以依照自己的喜好来加以安装与分析你的 squid 登录档喔！鸟哥这里仅介绍一套相当强的分析软件， 那就是 sarg。

Squid Analysis Report Generator (Squid 分析报告制作者)，他的官方网站在：http://sarg.sourceforge.net/sarg.php，他的原理相当的简单，就是将 logfile 拿出来，然后进行一下解析，依据不同的时间、网站、与热门网站等等来进行数据的输出， 由于输出的结果实在是太详细了！所以...呵呵！如果你是老板的话，用这个软件会让你『爱不释手』啊！ 因为每个人的每个小动作都会被记录下来，我的天呐！当我第一次看到这个分析的画面时，真的给他吓了老大一跳得说～因为连每个 IP 在『每个小时所连上的每个网站数据』都有纪录～～害怕了吧～

不过，有优点就有缺点啦！怎么说呢？因为 SARG 功能太强大了，所以记录的『数据量』就实在是多了点，如果你的 Proxy 网站属于那种很大流量的网站时，那么就不要使用『日报表』，也就是每天产生一份报表的那种方式！ 那么由于数据一天可能会有几 MB 的数据，一两个月还没有关系，如果记录了几年，那么光是这些记录就会花掉好几 GB 的硬盘空间了～ 此外，也可以使用『覆盖旧有数据』的方式不要留存旧数据，这样也可以节省硬盘的空间啦！

在 SARG 的官网上面已经有朋友替大家将 RPM 的档案制作出来了，你可以参考：http://packages.sw.be/sarg/ 网站内的档案。由于鸟哥用的是 CentOS 6.x 64 位版本，但截至本日为止 (2011/08) 这个网站尚未释出稳定的 CentOS 6 版本，因此鸟哥下载的是 sarg-2.2.3.1-1.el5.rf.x86_64.rpm 这个版本。你可以使用 wget 下载到 /root 底下，再用 rpm -ivh 去安装起来即可。 这个软件默认会将 /var/www/sarg 作为输出报表的目标，而且你必须要安装与启动 WWW 服务器， 至于网址列则是： http://your.hostname/sarg 去查阅。底下让我们来处理 sarg 的配置文件吧！

```
[root@www ~]# yum install gd
[root@www ~]# rpm -ivh sarg-2.2.3.1-1.el5.rf.x86_64.rpm
[root@www ~]# vim /etc/sarg/sarg.conf
title "Squid 使用者存取报告"          &lt;==第 49 行左右
font_size 12px                         &lt;==第 69 行左右
charset UTF-8                          &lt;==第 353 行左右

# 1\. 一口气制作所有登录文件内的数据报表
[root@www ~]# sarg
SARG: Records in file: 2285, reading: 100.00%  &lt;==列出分析信息

# 2\. 制作 8 月 2 日的报表
[root@www ~]# sarg -d 02/08/2011
# 这两个范例，都会将数据丢到 /var/www/sarg/ONE-SHOT/ 底下去；

# 3\. 制作昨天的报表
[root@www ~]# sh /etc/cron.daily/sarg
# 这个范例则是将每天的数据放置于 /var/www/sarg/daily/ 底下去！
```

如果制作好相关数据，由于 sarg 这个 RPM 档案已经帮我们设定好了每日、每周、每月进行一次执行， 所以你可以不用管怎么执行啦！非常的方便！如果想要查阅数据，只要在 proxy server 端输入 http://your.hostname/sarg 会看到如下画面：

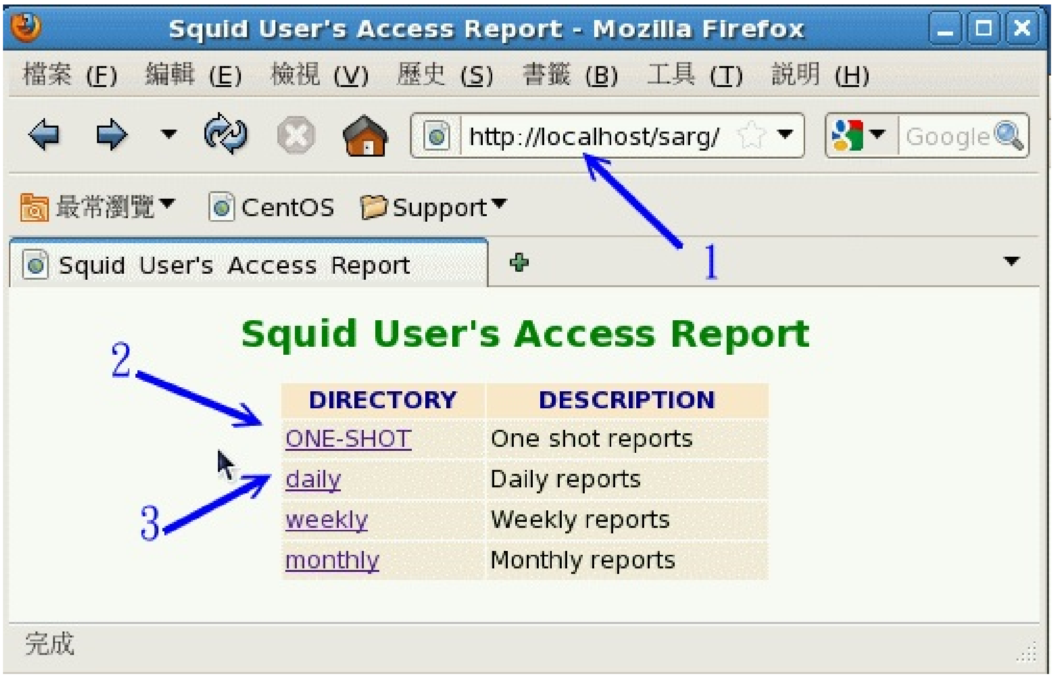


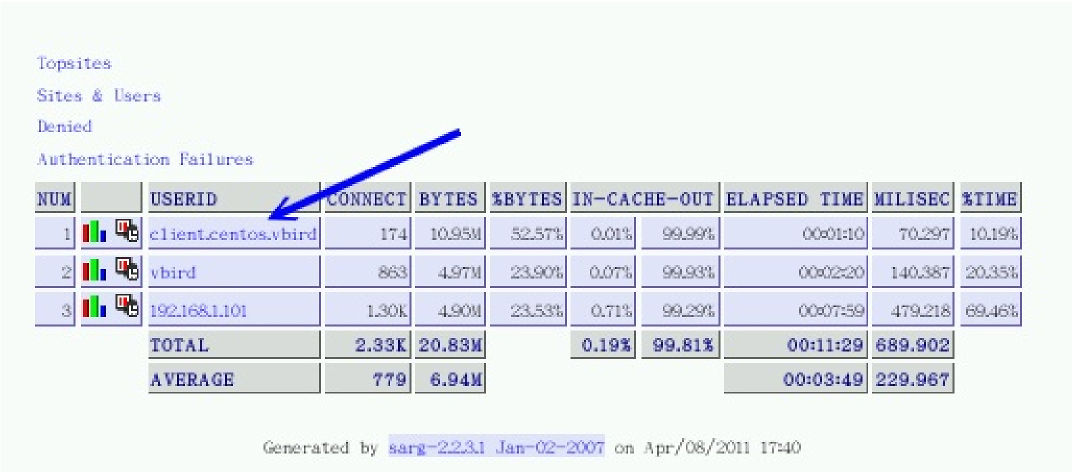
图 17.4-2、sarg 报表观察示意图

如上所示，在网址列输入服务器本机的咚咚，然后会看到几个连结。与我们有关的是 ONE-SHOT 以及 daily 两个， 我们来瞧瞧 ONE-SHOT (箭头 2 所指) 里面有啥咚咚？按下去会看到下图：



图 17.4-3、sarg 报表观察示意图

如上图所示，因为我们刚刚测试执行过两次 sarg 的指令，所以这里会有两个时间的连结。我们先看看总和数据， 亦即图中箭头所指的地方，会出现下图的说明：



Topsites
Sites & Users
Denied
Authentication Failures

NUM		USERID	CONNECT	BYTES	%BYTES	IN-CACHE-OUT		ELAPSED TIME	MILISEC	%TIME
1		client.centos.vbird	174	10.95M	52.57%	0.01%	99.99%	00:01:10	70.297	10.19%
2		vbird	863	4.97M	23.90%	0.07%	99.93%	00:02:20	140.387	20.35%
3		192.168.1.101	1.30K	4.90M	23.53%	0.71%	99.29%	00:07:59	479.218	69.46%
		TOTAL	2.33K	20.83M		0.19%	99.81%	00:11:29	689.902	
		AVERAGE	779	6.94M				00:03:49	229.967	

Generated by sarg-2.2.3.1 Jan-02-2007 on Apr/08/2011 17:40

图 17.4-4、sarg 报表观察示意图

在该段时间内，共有三个用户在存取，我们来瞧瞧 client.centos.vbird 到底干了啥事吧！



Squid 使用者存取報告
Period: 2011Apr07-2011Apr08
User: client.centos.vbird
Sort: BYTES, reverse
User Report

	ACCESSED SITE	CONNECT	BYTES	%BYTES	IN-CACHE-OUT		ELAPSED TIME	MILISEC	%TIME
	safebrowsing-cache.google.com	96	10.53M	96.17%	0.00%	100.00%	00:00:31	31.445	44.73%
	wiki.centos.org	22	172.31K	1.57%	0.00%	100.00%	00:00:15	15.328	21.80%
	www.centos.org	17	169.89K	1.55%	0.00%	100.00%	00:00:15	15.817	22.50%
	safebrowsing.clients.google.com	6	18.93K	0.17%	0.00%	100.00%	00:00:00	260	0.37%
	192.168.100.254	6	14.20K	0.13%	0.00%	100.00%	00:00:00	485	0.69%
	centos.tt.co.kr	4	10.16K	0.09%	0.00%	100.00%	00:00:02	2.222	3.16%
	ftp.daum.net	4	9.79K	0.09%	0.00%	100.00%	00:00:00	738	1.05%
	dic.vbird.tw	10	7.25K	0.07%	21.96%	78.04%	00:00:02	2.305	3.28%
	ftp.isu.edu.tw	3	6.01K	0.05%	0.00%	100.00%	00:00:00	50	0.07%
	data.nicehosting.co.kr	2	5.08K	0.05%	0.00%	100.00%	00:00:00	417	0.59%
	centos.mirror.cdnetworks.com	3	4.00K	0.04%	0.00%	100.00%	00:00:00	524	0.75%
	creativecommons.org	1	1.46K	0.01%	0.00%	100.00%	00:00:00	706	1.00%
	TOTAL	174	10.95M	52.57%	0.01%	99.99%	00:01:10	70.297	10.19%
	AVERAGE	779	6.94M				00:03:49	229.967	33.33%

Generated by sarg-2.2.3.1 Jan-02-2007 on Apr/08/2011 17:40

图 17.4-5、sarg 报表观察示意图

看到没有，这个用户在这段时间进行过的联机通通在里面！有没有很清晰呢？

17.5 重点回顾

- 代理服务器的功能是在代理用户向因特网要求 Web page 的数据，同时达成 Web pages 的快取记录，以达到带宽节省的目的； 此外，还可以额外的达成防火墙的功能；
- 我们可以透过具有较大带宽的上层代理服务器来进行捉取数据的分流；
- 设定 Proxy 时，如果能以带宽更大的上层 Proxy 来帮助，将有助于 Client 端浏览速度的提升；
- 以防火墙的功能来说， Proxy 使用应用层的方式来达成防火墙功能，至于 iptables 则是更为底层的 TCP/IP 分析的方式；
- 目前 Unix Like 的机器中，做为 proxy 功能的服务器软件几乎都是使用 squid ，而 squid 仅需要设定 squid.conf 这个配置文件即可使用；
- squid 主要透过 acl 配合 http_access 来进行信任用户与目标 WWW 服务器的控管；
- 用 http_access 这个参数来设定控管行为时，『顺序』是有影响的
- transparent proxy 的功能就是可以让 client 端不需要设定浏览器的 proxy 功能，即可进行 proxy 的工作；

17.6 本章习题

- 请说明为何 Proxy 可以提升网络的 WWW 浏览速度？
- 万一 squid 发生了问题，请问我该如何找出问题点？
- 请说明 Proxy 服务器的功能为何？
- 试说明为何 Proxy 服务器可以提升网域之内的网络安全性？

17.7 参考数据与延伸阅读

- squid 官方网站：http://www.squid-cache.org/
- squid 说明文件计划：http://squid-docs.sourceforge.net/, http://www.deckle.co.za/squid-users-guide/
- squid 的验证流程：http://www.l-penguin.idv.tw/article/proxy-auth.htm
- 旧版的一些范例参考：http://linux.vbird.org/linux_server/0420squid/0420squid_vbird_ex
- squid 官网收集的登录文件分析软件：http://www.squid-cache.org/Scripts/

2001/??/??：第一次完成日期，其实已经忘记了～ 2001/11/09：加入增加 Proxy 效能的方法，就是使用多颗硬盘做成的数据储存方式！ 2003/04/04：完成大幅度的改写动作！加入了完整的 Proxy 说明，与 pwebstats 的架设！ 2003/04/11：完成了另一个末端分析的强大软件 SARG 分析套件！ 2003/09/16：微幅调校一下版面！ 2004/11/12：修订 transparent proxy 的设定问题， httpd_accel_with_proxy on 2011/03/31：将旧的基于好老的 Red Hat 9 的文章移动到 此处 2011/04/08：累死了～这篇修改的幅度太大了！好疲倦～ 2011/08/02：将基于 CentOS 5.x 的版本移动到此处

第十八章、网络驱动器装置：iSCSI 服务器

最近更新日期：2011/08/02

如果你的系统需要大量的磁盘容量，但是身边却没有 NAS 或外接的储存设备，仅有个人计算机时，那该如何是好？ 此时，透过网络的 SCSI 磁盘 (iSCSI) 就能够有大大的帮助啦！这个 iSCSI 是将来自网络的数据仿真成本机的 SCSI 设备， 因此可以进行诸如 LVM 等方面的实作，而不是单纯使用服务器端提供的文件系统而已，相当的有帮助喔！

18.1 网络文件系统还是网络驱动器

做为服务器的系统通常是需要储存设备的，而储存设备除了可以使用系统内建的磁盘之外，如果内建的磁盘容量不够大， 而且也没有额外的磁盘插槽 (SATA 或 IDE) 可用时，那么常见解决的方案就是增加 NAS (网络附加储存服务器) 或外接式储存设备。再高档一点的系统，可能就会用到 SAN (储存局域网络) 这个高贵的玩意儿 (注1)。

不过，不论是哪一种架构，基本上，它们的内部硬盘通常是以磁盘阵列 (RAID) 作为基础的。磁盘阵列我们在基础篇第三版的第十五章谈过了，这里就不再啰唆。这里想要了解的是，啥是 NAS 又啥是 SAN ？ 这两者有何不同？与本章主题有关的 iSCSI 又是啥呢？底下就让我们来谈一谈。

18.1.1 NAS 与 SAN

由于企业的数据量越来越大，而且重要性与保密性越来越高，尤其类似数据库的内容，常常容量单位是以 TB (1TB = 1024GB) 在进行计算的。哇！真可怕！所以啰，磁盘阵列的应用就很重要了。那么磁盘阵列通常是在哪里呢？ 磁盘阵列通常是 (1)主机内部有磁盘阵列控制卡，可以自行管理磁盘阵列。不过想要提供磁盘阵列的容量， 得要透过额外的网络服务才行；(2)外接式磁盘阵列设备，就是单纯的磁盘阵列设备，必须透过某些接口链接到主机上， 主机也要安装适当的驱动程序后，才能捉到这个设备所提供的磁盘容量。

不过，以目前的信息社会来说，你应该很少听到内建或外接的 RAID 了，常常听到的应该是 NAS 与 SAN ，那这是啥咚咚？ 让我们简单的来说说：

- NAS (Network Attached Storage, 网络附加储存服务器)

基本上，NAS 其实就是一部客制化好的主机了，只要将 NAS 连接上网络，那么在网络上面的其他主机就能够存取 NAS 上头的资料了。简单的说，NAS 就是一部 file server 啰～不过，NAS 由于也是接在网络上面，所以，如果网络上有某个用户大量存取 NAS 上头的数据时，是很容易造成网络停顿的问题的，这个比较麻烦点～低阶的 NAS 通常会使用 Linux 系统搭配软件磁盘阵列来提供大容量文件系统。不过效能嘛就有待加强啦！此外，NAS 也通常支持 TCP/IP ，并会提供 NFS, SAMBA, FTP 等常见的通讯协议来提供客户端取得文件系统。

那为什么不要直接使用个人计算机安装 Linux 再搭配相关的服务，即可提供 NAS 预计要提供的大容量空间啦！干嘛需要 NAS 呢？ 因为，通常 NAS 还会包括很多组态的接口，通常是利用 Web 接口来控制磁盘阵列的设定状况、提供 IP 或其他相关网络设定， 以及是否提供某些特定的服务等等。因为具有较为亲和的操作与控制接口，对于非 IT 的人员来说，控管较为容易啦。 这也是 NAS 存在的目的。

不过，目前倒是有类似 FreeNAS 的软件开发项目 (http://sourceforge.net/projects/freenas/, 注2)，可以让你的 Linux PC 变成一部可透过 Web 控管的 NAS 哩！不过不是本章的重点，有兴趣的朋友可以自行前往下载与安装该软件来玩玩～

- SAN (Storage Area Networks, 储存局域网络)

从上面的说明来看，其实那个 NAS 就是一部可以提供大容量文件系统的主机嘛！ 那我们知道单部主机能够提供的插槽再怎么说也是有限的！ 所以并不能无限制的安插磁盘在同一部实体主机上面。但是如果偏偏你就是有大量磁盘使用的需求，那时该如何是好？ 这时就得要使用到 SAN 这玩意儿啦！

最简单的看法，就是将 SAN 视为一个外接式的储存设备。只是单纯的外接式储存设备仅能透过某些接口 (如 SCSI 或 eSATA) 提供单一部主机使用，而 SAN 却可以透过某些特殊的接口或信道来提供局域网络内的所有机器进行磁盘存取。要注意喔，SAN 是提供『磁盘 (block device)』给主机用，而不是像 NAS 提供的是『网络协议的文件系统 (NFS, SMB...)』！这两者的差异挺大的喔！因此，挂载使用 SAN 的主机会多出一个大磁盘，并可针对 SAN 提供的磁盘进行分割与格式化等动作。想想看，你能对 NAS 提供的文件系统格式化吗？不行吧！这样了解差异否？

另外，既然 SAN 可以提供磁盘，而 NAS 则是提供相关的网络文件系统，那么 NAS 能不能透过网络去使用 SAN 所提供的磁盘呢？答案当然是可以啊！因为 SAN 最大的目的就是在提供磁盘给服务器主机使用，NAS 也是一部完整的服务器， 所以 NAS 当然可以使用 SAN 啦！同时其他的网络服务器也能够使用这个 SAN 来进行数据存取。

此外，既然 SAN 开发的目的是要提供大量的磁盘给用户，那么传输的速度当然是非常重要的。因此，早期的 SAN 大多配合光纤信道 (Fibre Channel) 来提供高速的数据传输。目前标准的光纤信道是速度是 2GB ，未来还可能到达 10GB 以上呢～不过，使用光纤等技术较高的设备，当然就比较贵一些。

拜以太网络盛行，加上技术成熟之赐，现今的以太网络媒体 (网络卡、交换器、路由器等等设备) 已经可以达到 GB 的速度了，离 SAN 的光纤信道速度其实差异已经缩小很多啦～那么是否我们可以透过这个 GB 的以太网络接口来连接到 SAN 的设备呢？这就是我们接下来要提到的 iSCSI 架构啦！ ^_^

18.1.2 iSCSI 界面

早期的企业使用的服务器若有大容量磁盘的需求时，通常是透过 SCSI 来串接 SCSI 磁盘，因此服务器上面必须要加装 SCSI 适配卡，而且这个 SCSI 是专属于该服务器的。后来这个外接式的 SCSI 设备被上述提到的 SAN 的架构所取代， 在 SAN 的标准架构下，虽然有很多的服务器可以对同一个 SAN 进行存取的动作，不过为了速度需求，通常使用的是光纤信道。 但是光纤信道就是贵嘛！不但设备贵，服务器上面也要有光纤接口，很麻烦～所以光纤的 SAN 在中小企业很难普及啊～

后来网络实在太普及，尤其是以 IP 封包为基础的 LAN 技术已经很成熟，再加上以太网络的速度越来越快， 所以就有厂商将 SAN 的连接方式改为利用 IP 技术来处理。然后再透过一些标准的订定，最后就得到 Internet SCSI (iSCSI) 这玩意的产生啦！iSCSI 主要是透过 TCP/IP 的技术，将储存设备端透过 iSCSI target (iSCSI 目标) 功能，做成可以提供磁盘的服务器端，再透过 iSCSI initiator (iSCSI 初始化用户) 功能，做成能够挂载使用 iSCSI target 的客户端，如此便能透过 iSCSI 协议来进行磁盘的应用了 (注3)。

也就是说，iSCSI 这个架构主要将储存装置与使用的主机分为两个部分，分别是：

- iSCSI target：就是储存设备端，存放磁盘或 RAID 的设备，目前也能够将 Linux 主机仿真成 iSCSI target 了！目的在提供其他主机使用的『磁盘』；
- iSCSI initiator：就是能够使用 target 的客户端，通常是服务器。 也就是说，想要连接到 iSCSI target 的服务器，也必须要安装 iSCSI initiator 的相关功能后才能够使用 iSCSI target 提供的磁盘就是了。

如下图所示，iSCSI 是在 TCP/IP 上面所开发出来的一套应用，所以得要有网络才行啊！

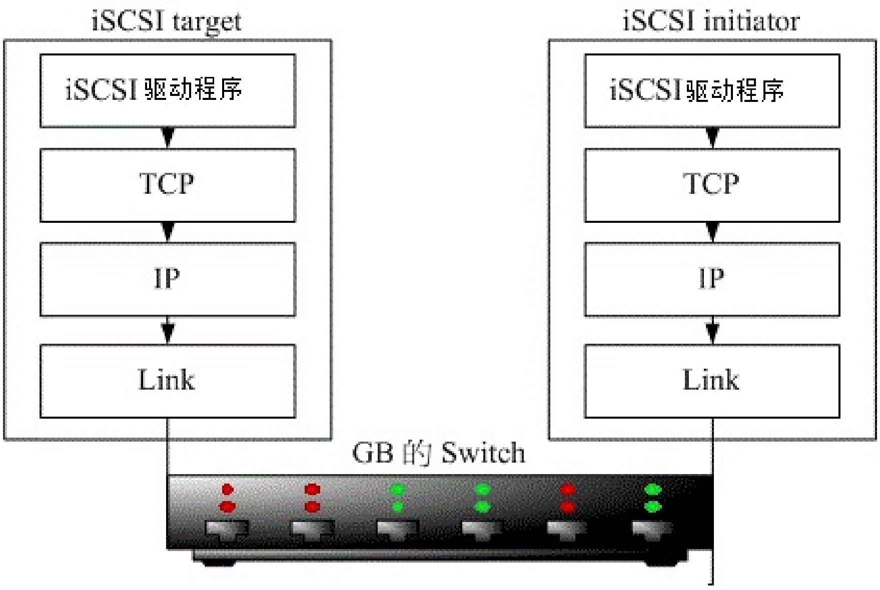


图 18.1-1、iSCSI 与 TCP/IP 相关性

18.1.3 各组件相关性

由上面的说明中，我们可以知道一部服务器如何取得磁盘或者是文件系统来利用呢？基本上就是：

- 直接存取 (direct-attached storage)：例如本机上面的磁盘，就是直接存取设备；
- 透过储存局域网络 (SAN)：来自区网内的其他储存设备提供的磁盘；
- 网络文件系统 (NAS)：来自 NAS 提供的文件系统，只能立即使用，不可进行格式化。

这三个东西与服务器主机能用的文件系统之间可以用维基百科的图示来展示：

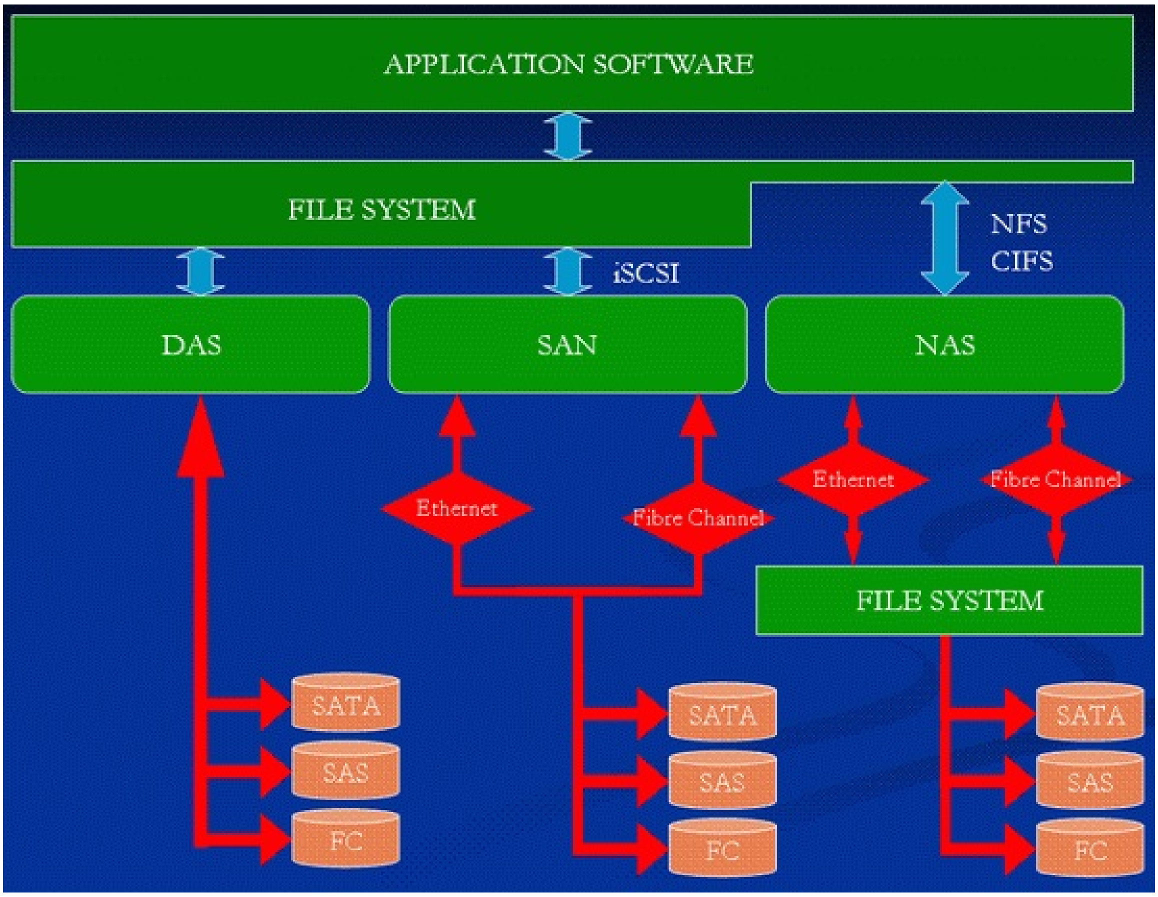


图 18.1-2、服务器取得文件系统的三个来源 (数据源为注1)

从上图中，我们可以发现在一般的主机环境下，磁盘装置 (SATA, SAS, FC) 可以透过主机的接口 (DAS) 来直接进行文件系统的建立 (mkfs 进行格式化)，如果想要使用外部的磁盘，那可以透过 SAN (内含多个磁盘的设备)，然后透过 iSCSI 等接口来联机， 当然，还是得要进行格式化等动作 (假设这个 SAN 尚未被使用时)。最后，如果是 NAS 的条件下，那么 NAS 必须要先透过自己的操作系统将磁盘装置进行文件系统的建立后，再以 NFS/CIFS 等方式来提供其他主机挂载使用。

接下来，网络服务器、客户端系统、NAS 与 SAN 的角色在区网里面又是如何呢？我们依旧使用维基百科的图示来说明一下 (DAS 是每部主机内部的磁盘，即底下图标中的圆柱体)：

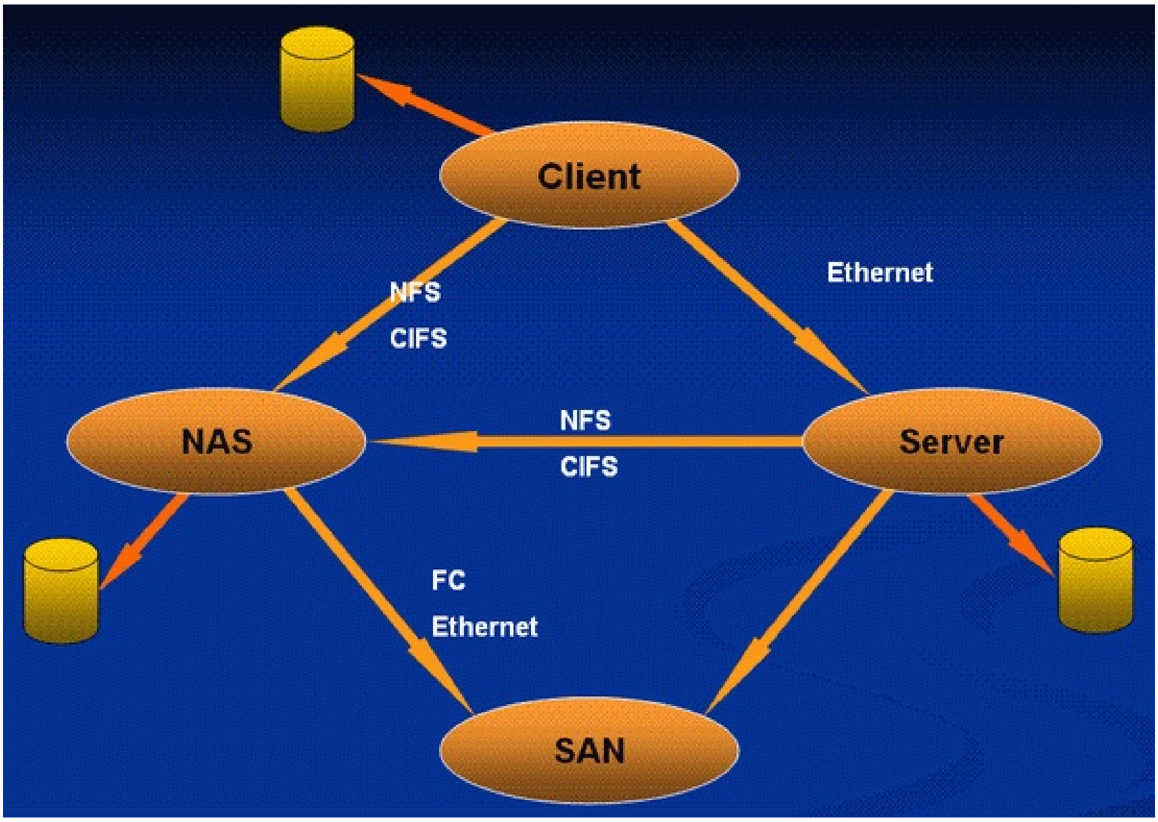


图 18.1-3、各组件之间的相关性 (数据源为注1)

NAS 可以使用自己的磁盘，也能够透过光纤或以太网络取得 SAN 所提供的磁盘来制作成为网络文件系统，提供其他人的使用。 Server 可以透过 NFS/CIFS 等方式取得 NAS 的文件系统，当然也能够直接存取 SAN 的磁盘。客户端主要则是透过网络文件系统， 并且直接使用 Server 提供的网络资源 (如 FTP, WWW, mail 等等)。

18.2 iSCSI target 的设定

能够完成 iSCSI target/initiator 设定的项目非常多 (注4)，鸟哥找的到的就有底下这几个：

- Linux SCSI target framework (tgt)：http://stgt.sourceforge.net/
- Linux-iSCSI Project：http://linux-iscsi.sourceforge.net/
- Open-iSCSI：http://www.open-iscsi.org/

由于被我们 CentOS 6.x 官方直接使用的是 tgt 这个软件，因此底下我们会使用 tgt 来介绍整个 iSCSI target 的设定喔！

18.2.1 所需软件与软件结构

CentOS 将 tgt 的软件名称定义为 scsi-target-utils ，因此你得要使用 yum 去安装他才行。至于用来作为 initiator 的软件则是使用 linux-iscsi 的项目，该项目所提供的软件名称则为 iscsi-initiator-utils 。所以，总的来说，你需要的软件有：

- scsi-target-utils：用来将 Linux 系统仿真成为 iSCSI target 的功能；
- iscsi-initiator-utils：挂载来自 target 的磁盘到 Linux 本机上。

那么 scsi-target-utils 主要提供哪些档案呢？基本上有底下几个比较重要需要注意的：

- /etc/tgt/targets.conf：主要配置文件，设定要分享的磁盘格式与哪几颗；
- /usr/sbin/tgt-admin：在线查询、删除 target 等功能的设定工具；
- /usr/sbin/tgt-setup-lun：建立 target 以及设定分享的磁盘与可使用的客户端等工具软件。
- /usr/sbin/tgtadm：手动直接管理的管理员工具 (可使用配置文件取代)；
- /usr/sbin/tgtd：主要提供 iSCSI target 服务的主程序；
- /usr/sbin/tgtimg：建置预计分享的映像文件装置的工具 (以映像文件仿真磁盘)；

其实 CentOS 已经将很多功能都设定好了，因此我们只要修订配置文件，然后启动 tgtd 这个服务就可以啰！ 接下来，就让我们实际来玩一玩 iSCSI target 的设定吧！

18.2.2 target 的实际设定

从上面的分析来看，iSCSI 就是透过一个网络接口，将既有的磁盘给分享出去就是了。那么有哪些类型的磁盘可以分享呢？ 这包括：

- 使用 dd 指令所建立的大型档案可供仿真为磁盘 (无须预先格式化)；
- 使用单一分割槽 (partition) 分享为磁盘；
- 使用单一完整的磁盘 (无须预先分割)；

- 使用磁盘阵列分享 (其实与单一磁盘相同方式)；
- 使用软件磁盘阵列 (software raid) 分享成单一磁盘；
- 使用 LVM 的 LV 装置分享为磁盘。

其实没有那么复杂，我们大概知道可以透过 (1)大型档案； (2)单一分割槽； (3)单一装置 (包括磁盘、数组、软件磁盘阵列、LVM 的 LV 装置文件名等等) 来进行分享。在本小节当中，我们将透过新的分割产生新的没有用到的分割槽、LVM 逻辑滚动条、大型档案等三个咚咚来进行分享。既然如此，那就得要先来搞定这些咚咚啰！ 要注意喔，等一下我们要分享出去的数据，最好不要被使用，也最好不要开机就被挂载 (/etc/fstab 当中没有存在记录的意思)。 那么就来玩玩看啰！

- 建立所需要的磁盘装置

既然 iSCSI 要分享的是磁盘，那么我们得要准备好啊！目前预计准备的磁盘为：

- 建立一个名为 /srv/iscsi/disk1.img 的 500MB 档案；
- 使用 /dev/sda10 提供 2GB 作为分享 (从第一章到目前为止的分割数)；
- 使用 /dev/server/iscsi01 的 2GB LV 作为分享 (再加入 5GB /dev/sda11 到 server VG 中)。

实际处理的方式如下：

```
# 1\. 建立大型档案：
[root@www ~]# mkdir /srv/iscsi
[root@www ~]# dd if=/dev/zero of=/srv/iscsi/disk1.img bs=1M count=500
[root@www ~]# chcon -Rv -t tgtd_var_lib_t /srv/iscsi/
[root@www ~]# ls -lh /srv/iscsi/disk1.img
-rw-r--r--. 1 root root 500M Aug  2 16:22 /srv/iscsi/disk1.img &lt;==容量对的！

# 2\. 建立实际的 partition 分割：
[root@www ~]# fdisk /dev/sda  &lt;==实际的分割方式自己处理吧！
[root@www ~]# partprobe       &lt;==某些情况下得 reboot 喔！
[root@www ~]# fdisk -l
   Device Boot      Start         End      Blocks   Id  System
/dev/sda10           2202        2463     2104483+  83  Linux
/dev/sda11           2464        3117     5253223+  8e  Linux LVM
# 只有输出 /dev/sda{10,11} 信息，其他的都省略了。注意看容量，上述容量单位 KB

[root@www ~]# swapon -s; mount &#124; grep 'sda1'
# 自己测试一下 /dev/sda{10,11} 不能够被使用喔！若有被使用，请 umount 或 swapoff

# 3\. 建立 LV 装置 ：
[root@www ~]# pvcreate /dev/sda11
[root@www ~]# vgextend server /dev/sda11
[root@www ~]# lvcreate -L 2G -n iscsi01 server
[root@www ~]# lvscan
  ACTIVE            '/dev/server/myhome' [6.88 GiB] inherit
  ACTIVE            '/dev/server/iscsi01' [2.00 GB] inherit
```

- 规划分享的 iSCSI target 檔名

iSCSI 有一套自己分享 target 档名的定义，基本上，藉由 iSCSI 分享出来的 target 檔名都是以 iqn 为开头，意思是：『iSCSI Qualified Name (iSCSI 合格名称)』的意思(注5)。那么在 iqn 后面要接啥档名呢？通常是这样的：

```
iqn.yyyy-mm.&lt;reversed domain name&gt;:identifier
iqn.年年-月.单位网域名的反转写法  :这个分享的target名称
```

鸟哥做这个测试的时间是 2011 年 8 月份，然后鸟哥的机器是 www.centos.vbird ，反转网域写法为 vbird.centos， 然后，鸟哥想要的 iSCSI target 名称是 vbirddisk ，那么就可以这样写：

- iqn.2011-08.vbird.centos:vbirddisk

另外，就如同一般外接式储存装置 (target 名称) 可以具有多个磁盘一样，我们的 target 也能够拥有数个磁盘装置的。 每个在同一个 target 上头的磁盘我们可以将它定义为逻辑单位编号 (Logical Unit Number, LUN)。我们的 iSCSI initiator 就是跟 target 协调后才取得 LUN 的存取权就是了 (注5)。在鸟哥的这个简单案例中，最终的结果，我们会有一个 target ，在这个 target 当中可以使用三个 LUN 的磁盘。

- 设定 tgt 的配置文件 /etc/tgt/targets.conf

接下来我们要开始来修改配置文件了。基本上，配置文件就是修改 /etc/tgt/targets.conf 啦。这个档案的内容可以改得很简单， 最重要的就是设定前一点规定的 iqn 名称，以及该名称所对应的装置，然后再给予一些可能会用到的参数而已。 多说无益，让我们实际来实作看看：

```
[root@www ~]# vim /etc/tgt/targets.conf
# 此档案的语法如下：
&lt;target iqn.相关装置的target名称&gt;
    backing-store /你的/虚拟设备/完整檔名-1
    backing-store /你的/虚拟设备/完整檔名-2
&lt;/target&gt;
 &lt;target iqn.2011-08.vbird.centos:vbirddisk&gt;
    backing-store /srv/iscsi/disk1.img  &lt;==LUN 1 (LUN 的编号通常照顺序)
    backing-store /dev/sda10            &lt;==LUN 2
    backing-store /dev/server/iscsi01   &lt;==LUN 3
&lt;/target&gt;
```

事实上，除了 backing-store 之外，在这个配置文件当中还有一些比较特别的参数可以讨论看看 (man tgt-admin)：

- backing-store (虚拟的装置), direct-store (实际的装置)： 设定装置时，如果你的整颗磁盘是全部被拿来当 iSCSI 分享之用，那么才能够使用 direct-store 。不过，根据网络上的其他文件， 似乎说明这个设定值有点危险的样子。所以，基本上还是建议单纯使用模拟的 backing-store 较佳。例如鸟哥的简单案例中，就通通使用 backing-store 而已。

- initiator-address (用户端地址)： 如果你想要限制能够使用这个 target 的客户端来源，才需要填写这个设定值。基本上，不用设定它 (代表所有人都能使用的意思)， 因为我们后来会使用 iptables 来规范可以联机的客户端嘛！

- incominguser (用户账号密码设定)： 如果除了来源 IP 的限制之外，你还想要让使用者输入账密才能使用你的 iSCSI target 的话，那么就加用这个设定项目。 此设定后面接两个参数，分别是账号与密码啰。

- write-cache [off|on] (是否使用快取)： 在预设的情况下，tgtd 会使用快取来增快速度。不过，这样可能会有遗失数据的风险。所以，如果你的数据比较重要的话， 或许不要使用快取，直接存取装置会比较妥当一些。

上面的设定值要怎么用呢？现在，假设你的环境中，仅允许 192.168.100.0/24 这个网段可以存取 iSCSI target，而且存取时需要帐密分别为 vbirduser, vbirdpasswd ，此外，不要使用快取，那么原本的配置文件之外，还得要加上这样的参数才行 (基本上，使用上述的设定即可，底下的设定是多加测试用的，不需要填入你的设定中)。

```
[root@www ~]# vim /etc/tgt/targets.conf
&lt;target iqn.2011-04.vbird.centos:vbirddisk&gt;
    backing-store /home/iscsi/disk1.img
    backing-store /dev/sda7
    backing-store /dev/server/iscsi01
 initiator-address 192.168.100.0/24
    incominguser vbirduser vbirdpasswd
    write-cache off
&lt;/target&gt;
```

- 启动 iSCSI target 以及观察相关端口与磁盘信息

再来则是启动、开机启动，以及观察 iSCSI target 所启动的埠口啰：

```
[root@www ~]# /etc/init.d/tgtd start
[root@www ~]# chkconfig tgtd on
[root@www ~]# netstat -tlunp &#124; grep tgt
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address   Foreign Address   State   PID/Program name
tcp        0      0 0.0.0.0:3260    0.0.0.0:*         LISTEN  26944/tgtd
tcp        0      0 :::3260         :::*              LISTEN  26944/tgtd
# 重点就是那个 3260 TCP 封包啦！等一下的防火墙务必要开放这个埠口。

# 观察一下我们 target 相关信息，以及提供的 LUN 数据内容：
[root@www ~]# tgt-admin --show
Target 1: iqn.2011-08.vbird.centos:vbirddisk &lt;==就是我们的 target
    System information:
        Driver: iscsi
        State: ready
    I_T nexus information:
    LUN information:
        LUN: 0
            Type: controller     &lt;==这是个控制器，并非可以用的 LUN 喔！
....(中间省略)....
        LUN: 1
            Type: disk       &lt;==第一个 LUN，是磁盘 (disk) 喔！
            SCSI ID: IET     00010001
            SCSI SN: beaf11
            Size: 2155 MB      &lt;==容量有这么大！
            Online: Yes
            Removable media: No
            Backing store type: rdwr
            Backing store path: /dev/sda10 &lt;==磁盘所在的实际文件名
        LUN: 2
            Type: disk
            SCSI ID: IET     00010002
            SCSI SN: beaf12
            Size: 2147 MB
            Online: Yes
            Removable media: No
            Backing store type: rdwr
            Backing store path: /dev/server/iscsi01
        LUN: 3
            Type: disk
            SCSI ID: IET     00010003
            SCSI SN: beaf13
            Size: 524 MB
            Online: Yes
            Removable media: No
            Backing store type: rdwr
            Backing store path: /srv/iscsi/disk1.img
    Account information:
 vbirduser        &lt;==额外的帐户信息
    ACL information:
 192.168.100.0/24 &lt;==额外的来源 IP 限制
```

请将上面的信息对照一下我们的配置文件吻！看看有没有错误就是了！尤其注意每个 LUN 的容量、实际磁盘路径！ 那个项目不能错误就是了。(照理说 LUN 的数字应该与 backing-store 设定的顺序有关，不过，在鸟哥的测试中， 出现的顺序并不相同！因此，还是需要使用 tgt-admin --show 去查阅查阅才好！)

- 设定防火墙

不论你有没有使用 initiator-address 在 targets.conf 配置文件中，iSCSI target 就是使用 TCP/IP 传输数据的， 所以你还是得要在防火墙内设定可以联机的客户端才行！既然 iSCSI 仅开启 3260 埠口，那么我们就这么进行即可：

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.allow
iptables -A INPUT  -p tcp -s 192.168.100.0/24 --dport 3260 -j ACCEPT

[root@www ~]# /usr/local/virus/iptables/iptables.rule
[root@www ~]# iptables-save &#124; grep 3260
-A INPUT -s 192.168.100.0/24 -p tcp -m tcp --dport 3260 -j ACCEPT
# 最终要看到上述的输出字样才是 OK 的呦！若有其他用户需要联机，
# 自行复制 iptables.allow 内的语法，修改来源端即可。
```

18.3 iSCSI initiator 的设定

谈完了 target 的设定，并且观察到相关 target 的 LUN 数据后，接下来就是要来挂载使用啰。使用的方法很简单， 只不过我们得要安装额外的软件来取得 target 的 LUN 使用权就是了。

18.3.1 所需软件与软件结构

在前一小节就谈过了，要设定 iSCSI initiator 必须要安装 iscsi-initiator-utils 才行。安装的方法请使用 yum 去处理，这里不再多讲话。那么这个软件的结构是如何呢？

- /etc/iscsi/iscsid.conf：主要的配置文件，用来连结到 iSCSI target 的设定；
- /sbin/iscsid：启动 iSCSI initiator 的主要服务程序；
- /sbin/iscsiadm：用来管理 iSCSI initiator 的主要设定程序；
- /etc/init.d/iscsid：让本机模拟成为 iSCSI initiater 的主要服务；
- /etc/init.d/iscsi：在本机成为 iSCSI initiator 之后，启动此脚本，让我们可以登入 iSCSI target。所以 iscsid 先启动后，才能启动这个服务。为了防呆，所以 /etc/init.d/iscsi 已经写了一个启动指令， 启动 iscsi 前尚未启动 iscsid ，则会先呼叫 iscsid 才继续处理 iscsi 喔！

老实说，因为 /etc/init.d/iscsi 脚本已经包含了启动 /etc/init.d/iscsid 的步骤在里面，所以，理论上， 你只要启动 iscsi 就好啦！此外，那个 iscsid.conf 里面大概只要设定好登入 target 时的帐密即可， 其他的 target 搜寻、设定、取得的方法都直接使用 iscsiadm 这个指令来完成。由于 iscsiadm 侦测到的结果会直接写入 /var/lib/iscsi/nodes/ 当中，因此只要启动 /etc/init.d/iscsi 就能够在下次开机时，自动的连结到正确的 target 啰。 那么就让我们来处理处理整个过程吧 (注6)！

18.3.2 initiator 的实际设定

首先，我们得要知道 target 提供了啥咚咚啊，因此，理论上，不论是 target 还是 initiator 都应该是要我们管理的机器才对。 而现在我们知道 target 其实有设定账号与密码的，所以底下我们就得要修改一下 iscsid.conf 的内容才行。

- 修改 /etc/iscsi/iscsid.conf 内容，并启动 iscsi

这个档案的修改很简单，因为里面的参数大多已经预设做的不错了，所以只要填写 target 登入时所需要的帐密即可。 修改的地方有两个，一个是侦测时 (discovery) 可能会用到的帐密，一个是联机时 (node) 会用到的帐密：

```
[root@clientlinux ~]# vim /etc/iscsi/iscsid.conf
node.session.auth.username = vbirduser   &lt;==在 target 时设定的
node.session.auth.password = vbirdpasswd &lt;==约在 53, 54 行
discovery.sendtargets.auth.username = vbirduser  &lt;==约在 67, 68 行
discovery.sendtargets.auth.password = vbirdpasswd

[root@clientlinux ~]# chkconfig iscsid on
[root@clientlinux ~]# chkconfig iscsi on
```

由于我们尚未与 target 联机，所以 iscsi 并无法让我们顺利启动的！因此上面只要 chkconfig 即可，不需要启动他。 要开始来侦测 target 与写入系统信息啰。全部使用 iscsiadm 这个指令就可以完成所有动作了。

- 侦测 192.168.100.254 这部 target 的相关数据

虽然我们已经知道 target 的名字，不过，这里假设还不知道啦！因为有可能哪一天你的公司有钱了， 会去买实体的 iSCSI 数组嘛！所以这里还是讲完整的侦测过程好了！你可以这样使用：

```
[root@clientlinux ~]# iscsiadm -m discovery -t sendtargets -p IP:port
选项与参数：
-m discovery    ：使用侦测的方式进行 iscsiadmin 指令功能；
-t sendtargets ：透过 iscsi 的协议，侦测后面的设备所拥有的 target 数据
-p IP:port      ：就是那部 iscsi 设备的 IP 与埠口，不写埠口预设是 3260 啰！

范例：侦测 192.168.100.254 这部 iSCSI 设备的相关数据
[root@clientlinux ~]# iscsiadm -m discovery -t sendtargets -p 192.168.100.254
192.168.100.254:3260,1  iqn.2011-08.vbird.centos:vbirddisk
# 192.168.100.254:3260,1 ：在此 IP, 端口上面的 target 号码，本例中为 target1
# iqn.2011-08.vbird.centos:vbirddisk ：就是我们的 target 名称啊！

[root@clientlinux ~]# ll -R /var/lib/iscsi/nodes/
/var/lib/iscsi/nodes/iqn.2011-08.vbird.centos:vbirddisk
/var/lib/iscsi/nodes/iqn.2011-08.vbird.centos:vbirddisk/192.168.100.254,3260,1
# 上面的特殊字体部分，就是我们利用 iscsiadm 侦测到的 target 结果！
```

现在我们知道了 target 的名称，同时将所有侦测到的信息通通写入到上述 /var/lib/iscsi/nodes/iqn.2011-08.vbird.centos:vbirddisk/192.168.100.254,3260,1 目录内的 default 档案中， 若信息有修订过的话，那你可以到这个档案内修改，也可以透过 iscsiadm 的 update 功能处理相关参数的。

- 开始进行联机 iSCSI target

因为我们的 initiator 可能会连接多部的 target 设备，因此，我们得先要瞧瞧目前系统上面侦测到的 target 有几部， 然后再找到我们要的那部 target 来进行登入的作业。不过，如果你想要将所有侦测到的 target 全部都登入的话， 那么整个步骤可以再简化：

```
范例：根据前一个步骤侦测到的资料，启动全部的 target
[root@clientlinux ~]# /etc/init.d/iscsi restart
正在停止 iscsi：                                     [  确定  ]
正在激活 iscsi：                                     [  确定  ]
# 将系统里面全部的 target 通通以 /var/lib/iscs/nodes/ 内的设定登入
# 上面的特殊字体比较需要注意啦！你只要做到这里即可，底下的瞧瞧就好。

范例：显示出目前系统上面所有的 target 数据：
[root@clientlinux ~]# iscsiadm -m node
192.168.100.254:3260,1 iqn.2011-08.vbird.centos:vbirddisk
选项与参数：
-m node：找出目前本机上面所有侦测到的 target 信息，可能并未登入喔

范例：仅登入某部 target ，不要重新启动 iscsi 服务
[root@clientlinux ~]# iscsiadm -m node -T target名称 --login
选项与参数：
-T target名称：仅使用后面接的那部 target ，target 名称可用上个指令查到！
--login      ：就是登入啊！

[root@clientlinux ~]# iscsiadm -m node -T iqn.2011-08.vbird.centos:vbirddisk \
&gt;  --login
# 这次进行会出现错误，是因为我们已经登入了，不可重复登入喔！
```

接下来呢？呵呵！很棒的是，我们要来开始处理这个 iSCSI 的磁盘了喔！怎么处理？瞧一瞧！

```
[root@clientlinux ~]# fdisk -l
Disk /dev/sda: 8589 MB, 8589934592 bytes  &lt;==这是原有的那颗磁盘，略过不看
....(中间省略)....

Disk /dev/sdc: 2147 MB, 2147483648 bytes
67 heads, 62 sectors/track, 1009 cylinders
Units = cylinders of 4154 * 512 = 2126848 bytes
Sector size (logical/physical): 512 bytes / 512 bytes

Disk /dev/sdb: 2154 MB, 2154991104 bytes
67 heads, 62 sectors/track, 1013 cylinders
Units = cylinders of 4154 * 512 = 2126848 bytes
Sector size (logical/physical): 512 bytes / 512 bytes

Disk /dev/sdd: 524 MB, 524288000 bytes
17 heads, 59 sectors/track, 1020 cylinders
Units = cylinders of 1003 * 512 = 513536 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
```

你会发现主机上面多出了三个新的磁盘，容量与刚刚在 192.168.100.254 那部 iSCSI target 上面分享的 LUN 一样大。 那这三颗磁盘可以怎么用？你想怎么用就怎么用啊！只是，唯一要注意的，就是 iSCSI target 每次都要比 iSCSI initiator 这部主机还要早开机，否则我们的 initiator 恐怕就会出问题。

- 更新/删除/新增 target 数据的方法

如果你的 iSCSI target 可能因为某些原因被拿走了，或者是已经不存在于你的区网中，或者是要送修了～ 这个时候你的 iSCSI initiator 总是得要关闭吧！但是，又不能全部关掉 (/etc/init.d/iscsi stop)， 因为还有其他的 iSCSI target 在使用。这个时候该如何取消不要的 target 呢？很简单！流程如下：

```
[root@clientlinux ~]# iscsiadm -m node -T targetname --logout
[root@clientlinux ~]# iscsiadm -m node -o [delete&#124;new&#124;update] -T targetname
选项与参数：
--logout ：就是注销 target，但是并没有删除 /var/lib/iscsi/nodes/ 内的数据
-o delete：删除后面接的那部 target 链接信息 (/var/lib/iscsi/nodes/*)
-o update：更新相关的信息
-o new   ：增加一个新的 target 信息。

范例：关闭来自鸟哥的 iSCSI target 的数据，并且移除链接
[root@clientlinux ~]# iscsiadm -m node   &lt;==还是先秀出相关的 target iqn 名称
192.168.100.254:3260,1 iqn.2011-08.vbird.centos:vbirddisk
[root@clientlinux ~]# iscsiadm -m node -T &lt;u&gt;iqn.2011-08.vbird.centos:vbirddisk&lt;
&gt;  --logout
Logging out of session [sid: 1, target: iqn.2011-08.vbird.centos:vbirddisk,
 portal: 192.168.100.254,3260]
Logout of [sid: 1, target: iqn.2011-08.vbird.centos:vbirddisk, portal:
 192.168.100.254,3260] successful.
# 这个时候的 target 连结还是存在的，虽然注销你还是看的到！

[root@clientlinux ~]# iscsiadm -m node -o delete \
&gt;  -T iqn.2011-08.vbird.centos:vbirddisk
[root@clientlinux ~]# iscsiadm -m node
iscsiadm: no records found! &lt;==嘿嘿！不存在这个 target 了～

[root@clientlinux ~]# /etc/init.d/iscsi restart
# 你会发现唔！怎么 target 的信息不见了！这样瞭了乎！
```

如果一切都没有问题，现在，请回到 discovery 的过程，重新再将 iSCSI target 侦测一次，再重新启动 initiator 来取得那三个磁盘吧！我们要来测试与利用该磁盘啰！

18.3.3 一个测试范例

到底 iSCSI 可以怎么用？我们就来玩一玩。假设：

1. 你刚刚如同鸟哥的整个运作流程，已经在 initiator 上面将 target 数据清除了；
2. 现在我们只知道 iSCSI target 的 IP 是 192.168.100.254 ，而需要的帐密是 vbirduser, vbirdpasswd；
3. 帐密信息你已经写入 /etc/iscsi/iscsid.conf 里面了；
4. 假设我们预计要将 target 的磁盘拿来当作 LVM 内的 PV 使用；
5. 并且将所有的磁盘容量都给一个名为 /dev/iscsi/disk 的 LV 使用；
6. 这个 LV 会被格式化为 ext4 ，且挂载在 /data/iscsi 内。

那么，整体的流程是：

```
# 1\. 启动 iscsi ，并且开始侦测及登入 192.168.100.254 上面的 target 名称
[root@clientlinux ~]# /etc/init.d/iscsi restart
[root@clientlinux ~]# chkconfig iscsi on
[root@clientlinux ~]# iscsiadm -m discovery -t sendtargets -p 192.168.100.254
[root@clientlinux ~]# /etc/init.d/iscsi restart
[root@clientlinux ~]# iscsiadm -m node
192.168.100.254:3260,1 iqn.2011-08.vbird.centos:vbirddisk

# 2\. 开始处理 LVM 的流程，由 PV, VG, LV 依序处理喔！
[root@clientlinux ~]# fdisk -l    &lt;==出现的资料中你会发现 /dev/sd[b-d]
[root@clientlinux ~]# pvcreate /dev/sd{b,c,d}  &lt;==建立 PV 去！
  Wiping swap signature on /dev/sdb
  Physical volume "/dev/sdb" successfully created
  Physical volume "/dev/sdc" successfully created
  Physical volume "/dev/sdd" successfully created

[root@clientlinux ~]# vgcreate iscsi /dev/sd{b,c,d}  &lt;==建立 VG 去！
  Volume group "iscsi" successfully created

[root@clientlinux ~]# vgdisplay  &lt;==要找到可用的容量啰！
  --- Volume group ---
  VG Name               iscsi
....(中间省略)....
  Act PV                3
  VG Size               4.48 GiB
  PE Size               4.00 MiB
  Total PE              1148  &lt;==就是这玩意儿！共 1148 个！
  Alloc PE / Size       0 / 0
  Free  PE / Size       1148 / 4.48 GiB
....(底下省略)....

[root@clientlinux ~]# lvcreate -l 1148 -n disk iscsi
  Logical volume "disk" created

[root@clientlinux ~]# lvdisplay
  --- Logical volume ---
 LV Name                /dev/iscsi/disk
  VG Name                iscsi
  LV UUID                opR64B-Zeoe-C58n-ipN2-em3O-nUYs-wjEZDP
  LV Write Access        read/write
  LV Status              available
  # open                 0
  LV Size                4.48 GiB &lt;==注意一下容量对不对啊！
  Current LE             1148
  Segments               3
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:2

# 3\. 开始格式化，并且进行开机自动挂载的动作！
[root@clientlinux ~]# mkfs -t ext4 /dev/iscsi/disk
[root@clientlinux ~]# mkdir -p /data/iscsi
[root@clientlinux ~]# vim /etc/fstab
/dev/iscsi/disk   /data/iscsi   ext4   defaults,_netdev   1   2

[root@clientlinux ~]# mount -a
[root@clientlinux ~]# df -Th
文件系统      类型    Size  Used Avail Use% 挂载点
/dev/mapper/iscsi-disk
              ext4    4.5G  137M  4.1G   4% /data/iscsi
```

比较特殊的是 /etc/fstab 里面的第四个字段，加上 *netdev (*最前面是底线*)* 指的是，因为这个 *partition* 位于网络上， 所以得要网络开机启动完成后才会挂载的意思。现在，请让你的 *iSCSI initiator* 重新启动看看， 试看看重新启动系统后，你的 */data/iscsi* 是否还存在呢？ ^^

然后，让我们切回 iSCSI target 那部主机，研究看看到底谁有使用我们的 target 呢？

```
[root@www ~]# tgt-admin --show
Target 1: iqn.2011-08.vbird.centos:vbirddisk
    System information:
        Driver: iscsi
        State: ready
    I_T nexus information:
        I_T nexus: 2
            Initiator: iqn.1994-05.com.redhat:71cf137f58f2 &lt;==不是很喜欢的名字！
            Connection: 0
                IP Address: 192.168.100.10    &lt;==就是这里联机进来啰！
    LUN information:
....(后面省略)....
```

明明是 initiator 怎么会是那个 redhat 的名字呢？如果你不介意那就算了，如果挺介意的话，那么修改 initiator 那部主机的 /etc/iscsi/initiatorname.iscsi 这个档案的内容，将它变成类似如下的模样即可：

Tips: 不过，这个动作最好在使用 target 的 LUN 之前就进行，否则，当你使用了 LUN 的磁盘后，再修改这个档案后， 你的磁盘文件名可能会改变。例如鸟哥的案例中，改过 initiatorname 之后，原本的磁盘文件名竟变成 /dev/sd[efg] 了！害鸟哥的 LV 就不能再度使用了...




```
# 1\. 先在 iSCSI initiator 上面进行如下动作：
[root@clientlinux ~]# vim /etc/iscsi/initiatorname.iscsi
InitiatorName=iqn.2011-08.vbird.centos:initiator
[root@clientlinux ~]# /etc/init.d/iscsi restart

# 2\. 在 iSCSI target 上面就可以发现如下的数据修订了：
[root@www ~]# tgt-admin --show
Target 1: iqn.2011-08.vbird.centos:vbirddisk
    System information:
        Driver: iscsi
        State: ready
    I_T nexus information:
        I_T nexus: 5
 Initiator: iqn.2011-08.vbird.centos:initiator
            Connection: 0
                IP Address: 192.168.100.10
....(后面省略)....
```

18.4 重点回顾

- 如果需要大容量的磁盘，通常会使用 RAID 磁盘阵列的架构；
- 取得外部磁盘容量的作法，主要有 NAT 及 SAN 两大类的方式；
- NAT 可以想成是一部已经客制化的服务器，主要提供 NFS, SMB 等网络文件系统；
- SAN 则是一种外接是储存设备，可以透过 SAN 取得外部的磁盘装置 (非文件系统)；
- SAN 早期使用光纤信道，由于以太网络的发展，近来使用 iSCSI 协议在 TCP/IP 架构上面实作；
- iSCSI 协议主要分为 iSCSI target (提供磁盘装置者) 及 iSCSI initiator (存取 target 磁盘)；
- iSCSI target 主要使用 scsi-target-utils 软件达成主要利用 tgt-admin 及 tgtadm 指令完成：
- 一般定义 target 名称为：iqn.yyyy-mm.<reversed domain name>:identifier
- 一部 target 里面可分享多个磁盘，每个磁盘都是一个 LUN；
- iSCSI initiator 主要透过 iscsi-initiator-utils 软件达成链接到 target 的任务；
- iscsi-initiator-utils 主要提供 iscsiadm 来完成所有的动作。

18.5 本章习题

- 由于网络驱动器机的运作是需要很好的网络质量才行，我们这里仅在测试，因此，请将 client 端的 initiator 关闭， 否则，未来开机都会怪怪的！(chkconfig iscsi off; vim /etc/fstab 等等的动作！)

18.6 参考数据与延伸阅读

- 注1：SAN 与 NAS 在维基百科：http://en.wikipedia.org/wiki/Storage_area_network
- 注2：FreeNAS 的官网：http://sourceforge.net/projects/freenas/
- 注3：鸟站网友彦明兄对 iSCSI 的说明文件：http://linux.vbird.org/somepaper/20081205-rhel4-iscsi.pdf
- 注4：几个常见的将 Linux 模拟成 iSCSI target 与 initiator 的官网： Linux SCSI target framework (tgt)：http://stgt.sourceforge.net/ Linux-iSCSI Project：http://linux-iscsi.sourceforge.net/ Open-iSCSI：http://www.open-iscsi.org/
- 注5：iSCSI 内的 iqn 及 LUN 意义说明：http://en.wikipedia.org/wiki/ISCSI
- 注6：鸟站之友彦明兄提供的良好文献，以及相关的 initiator 设定方式：http://linux.vbird.org/somepaper/20081205-rhel5-iscsi.pdf iSCSI (client) howto：http://www.cyberciti.biz/tips/rhel-centos-fedora-linux-iscsi-howto.html 鸟站旧版资料：http://linux.vbird.org/linux_basic/0610hardware/0610hardware-fc4.php#raid_iscsi
- http://rhev-wiki.org/index.php?title=RHEL_5.5/CentOS_5.5_iSCSI_Storage_Server

---

2011/04/08：重新编辑本数据哩！ 2011/04/25：历经多个礼拜的杂事杂务缠身，终于完成这篇 iSCSI 的模拟应用。应该是挺好玩的一个咚咚～ 2011/08/02：将基于 CentOS 5.x 的版本移动到此处

第四部分：常见因特网服务器架设

讲到因特网服务器，你第一个会想到的应该就是 WWW 还有 FTP 吧！但其实还有一个更重要的你可能会不知道他的存在， 那就是 DNS ！这才是重点中的重点～因为我们都是使用主机名来联机的嘛！这部份一定会使用到 DNS 服务器，因此我们当然要了解一下。 最后再跟大家报告邮件服务器！这些服务器的设定以及未来的应用是非常好玩的，不过如果最前面的两篇您没有预先读过， 那么您的服务器被入侵也是一个可以『预期』的后果！所以啰，看完前两篇后，仔细的开始瞧一瞧这一篇的服务器吧！ ^_^

第十九章、主机名控制者：DNS 服务器

最近更新日期：2011/08/05

我们都知道，在『记忆』的角色上，人脑总是不如计算机的，而人们对文字的印象又比数字高。因此，想要使用纯粹的 TCP/IP 来上网，实在不好记忆又很麻烦。为了适应人类的使用习惯，因此一个名为 DNS 的服务，帮我们将主机名解析为 IP 好让大家只要记得主机名就能使用 Internet 的咚咚就这么诞生啦！在这一章当中，我们会谈一谈 DNS 服务内的正、反解 zone 的意义，解析主机名的授权概念与整体查询流程，以及 master/slave DNS 服务的配置等等吻！赶紧动动脑先～

19.1 什么是 DNS

DNS 越来越重要，尤其未来 IPv6 这个需要 128bits 地址的玩意儿。因为我们连 IPv4 的 32bits 都背不起来了， 128bits 要怎么背？ 这时主机名自动解析为 IP 就很重要啦！那就是 DNS。但是 DNS 的架设有点麻烦，重点是原理的部分比较不好理解。 因此在这个小节当中，让我们先来谈谈与网络主机名有关的一些知识，这样架设 DNS 才不会出问题。

19.1.1 用网络主机名取得 IP 的历史渊源

目前的因特网世界使用的是所谓的 TCP/IP 协议，其中 IP 为第四版的 IPv4 。不过，这个 IPv4 是由 32 位所组成，为了人脑已经转成四组十进制的数字了，例如 12.34.56.78 这样的格式。当我们利用 Internet 传送数据的时候，就需要这个 IP ，否则数据封包怎么知道要被送到哪里去？

- 单一档案处理上网的年代： /etc/hosts

然而人脑对于 IP 这种数字的玩意儿，记忆力实在是不怎么样。但是要上 Internet 又一定需要 IP，怎么办？为了应付这个问题， 早期的朋友想到一个方法，那就是利用某些特定的档案将主机名与 IP 作一个对应， 如此一来，我们就可以透过主机名来取得该主机的 IP 了！真是个好主意，因为人类对于名字的记忆力可就好多了！ 那就是 /etc/hosts 这个档案的用途了。

可惜的是，这个方法还是有缺憾的，那就是主机名与 IP 的对应无法自动于所有的计算机内更新， 且要将主机名加入该档案仅能向 INTERNIC 注册，若 IP 数量太多时，该档案会大到不象话，也就更不利于其他主机同步化了。 如下图所示，客户端计算机每次都得要重新下载一次档案才能顺利联网！

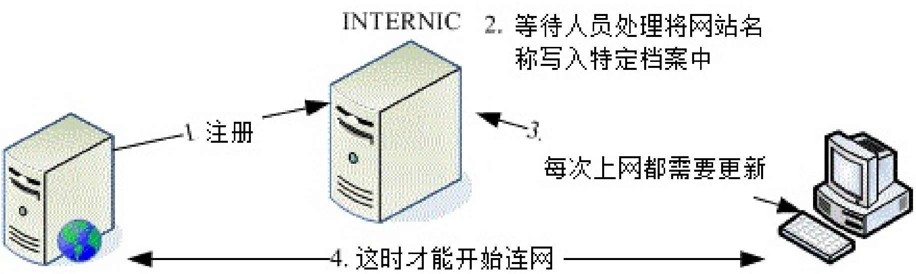


图 19.1-1、早期透过单一档案进行网络联机的示意图

在第四章 4.2.1 里面我们约略谈过 /etc/hosts 这个档案的用法，基本上该档案内容就是『IP 主机名 主机别名一 主机别名二...』 。在里面最重要的就是 localhost 对应到 127.0.0.1 这个咚咚！你千万不能删除该笔记录的。这里也再次强调，在你的私有网域内部，最好将所有的私有 IP 与主机名对应都写入这个档案中啦！

- 分布式、阶层式主机名管理架构： DNS 系统

早期网络尚未流行且计算机数量不多时，/etc/hosts 倒是还够用的，但自从 90 年代网络热门化后，单一档案 /etc/hosts 的联网问题就发生上面讲的状况啦！为了解决这个日益严重的问题，柏克莱大学发展出另外一套阶层式管理主机名对应 IP 的系统， 我们称它为 Berkeley Internet Name Domain, BIND ，这个系统可就优秀的多了～ 透过阶层式管理，可以轻松的进行维护的工作～太棒了！这也是目前全世界使用最广泛的领域名系统 (Domain Name System, DNS) 哩～透过 DNS ，我们不需要知道主机的 IP ，只要知道该主机的名称，就能够轻易的连上该主机了！

DNS 利用类似树状目录的架构，将主机名的管理分配在不同层级的 DNS 服务器当中，经由分层管理， 所以每一部 DNS 服务器记忆的信息就不会很多，而且若有 IP 异动时也相当容易修改！因为你如果已经申请到主机名解析的授权， 那么在你自己的 DNS 服务器中，就能够修改全世界都可以查询到的主机名了！而不用透过上层 ISP 的维护呢！ 自己动手当然是最快的啦！

由于目前的 IPv4 已经接近发送完毕的阶段，因此未来那个 128bits 的 IPv6 会逐渐热门起来。那么你需要背 128bits 的 IP 来上网吗？想必是不可能的！因此这个可以透过主机名就解析到 IP 的 DNS 服务，可以想象的到，它会越来越重要。此外，目前全世界的 WWW 主机名也都是透过 DNS 系统在处理 IP 的对应，所以，当 DNS 挂点时，我们将无法透过主机名来联机，那就几乎相当于没有 Internet 了！

因为 DNS 是这么的重要，所以即使我们没有架设它的必要时，还是得要熟悉一下它的原理才好。因此，跟 DNS 有关的 FQDN、Hostname 与 IP 的查询流程，正解与反解、合法授权的 DNS 服务器之意义，以及 Zone 等等的知识作一个认识才行！

Tips: 在底下的说明当中，我们有时会提到 DNS 有时会提到 BIND ，这有什么不同？ 由上面的说明里面，你可以了解到， DNS 是一种因特网的通讯协议名称， 至于 Bind 则是提供这个 DNS 服务的软件～这样你了解了吗？！



- 完整主机名： Fully Qualified Domain Name (FQDN)

第一个与 DNS 有关的主机名概念，就是『主机名与领域名 (hostname and domain name)』的观念，以及由这两者组成的完整主机名 Fully Qualified Domain Name, FQDN 的意义了。在讨论这个主题之前，我们来聊一聊比较生活化的话题：

- 以区域来区分同名同姓者的差异： 网络世界其实有很多人自称为『鸟哥』的，包括敝人在下小生我啦！那么你怎么知道此鸟哥非彼鸟哥呢？ 这个时候你可以利用每个鸟哥的所在地来作为区分啊，比如说台南的鸟哥与台北的鸟哥等。 那万一台南还有两个人自称鸟哥怎么办？没关系，你还可以依照乡镇来区分呢！比如说台南北区的鸟哥及台南中区的鸟哥。 如果将这个咚咚列出来，就有点像这样：

```
| 鸟哥、北区、台南 鸟哥、中区、台南 鸟哥、台北 ...... |
```

是否就可以分辨每个鸟哥的不同点了呢？呵呵！没错！就是这样！那个地区就是『领域 (domain) 』，而鸟哥就是主机名啦！

- 以区域号码来区分相同的电话号码： 另外一个例子可以使用电话号码来看，假如高雄有个 1234567 而台南也有个 1234567，那么(1)你在高雄直接拨接 1234567 时，他会直接挂入高雄的 1234567 电话中，(2)但如果你要拨到台南去，就得加入 (06) 这个区码才行！我们就是使用区码来做为辨识之用的！此时那个 06 区码就是 domain name，而电话号码就是主机名啦！

有没有一点点了解鸟哥想表达的啦？我们上面讲到，DNS 是以树状目录分阶层的方式来处理主机名，那我们知道树状目录中， 那个目录可以记录文件名。那么 DNS 记录的哪个咚咚跟『目录』有关？就是那个领域名。领域名底下还可以记录各个主机名， 组合起来才是完整的主机名 (FQDN)。

举例来说，我们常常会发现主机名都是 www 的网站，例如 www.google.com.tw, www.seednet.net, www.hinet.net 等等，那么我们怎么知道这些 www 名称的主机在不同的地方呢？就需要给他领域名啰！也就是 .google.com.tw, .seednet.net, .hinet.net 等等的不同，所以即使你的主机名相同，但是只要不是在同一个领域内，那么就可以被分辨出不同的位置啰！

我们知道目录树的最顶层是根目录 (/)，那么 DNS 既然也是阶层式的，最顶层是啥呢？每一层的 domain name 与 hostname 又该怎么分？我们举鸟哥所在的昆山科大的 WWW 服务器为例好了 (www.ksu.edu.tw) ：

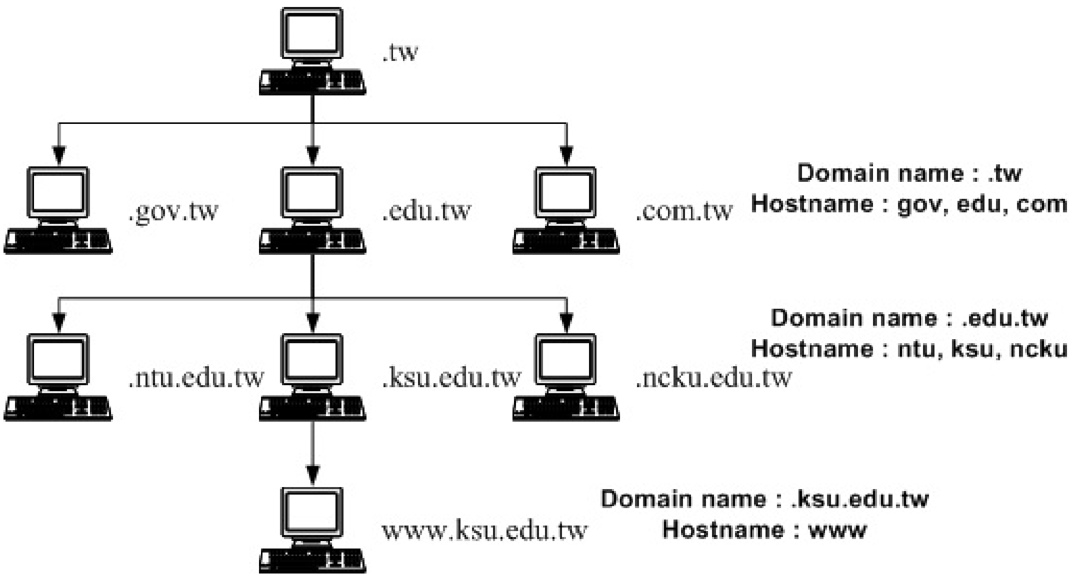


图 19.1-2、分阶层的 DNS 架构，以昆山科大为例 (hostname & domain name)

在上面的例子当中，由上向下数的第二层里面，那个 .tw 是 domain name ，而 com, edu, gov 则是主机的名称，而在这个主机的名称之管理下，还有其他更小网域的主机，所以在第三层的时候，基本上，那个 edu.tw 就变成了 domain name 了！而昆山科大与成大的 ksu, ncku

则成为了 hostname 啰！

以此类推，最后得到我们的主机那个 www 是主机名，而 domain name 是由 ksu.edu.tw 那个名字所决定的！自然，我们的主机就是让管理 ksu.edu.tw 这个 domain name 的 DNS 服务器所管理的啰！这样是否了解了 domain name 与 hostname 的不同了呢？

Tips: 并不是以小数点 (.) 区分 domain name 与 hostname 喔！某些时刻 domain name 所管理的 hostname 会含有小数点。 举例来说，鸟哥所在的信息传播系并没有额外的 DNS 服务器架设，因此我们的主机名为 www.dic ，而 domain name 还是 ksu.edu.tw ，因此全名为 www.dic.ksu.edu.tw 哩！



19.1.2 DNS 的主机名对应 IP 的查询流程

约略了解了 FQDN 的 domain name 与 hostname 之后，接下来我们要谈一谈这个 DNS 的：(1)阶层架构是怎样？ (2)查询原理是怎样？总是要先知道架构才能知道如何查询主机名的呐！所以底下我们先来介绍一下整体的 DNS 阶层架构。

- DNS 的阶层架构与 TLD

我们依旧使用台湾学术网络的 DNS 服务器所管理的各 domain 为例，将最上层到昆山科大 (ksu) 时，之间的各层绘制如下图：

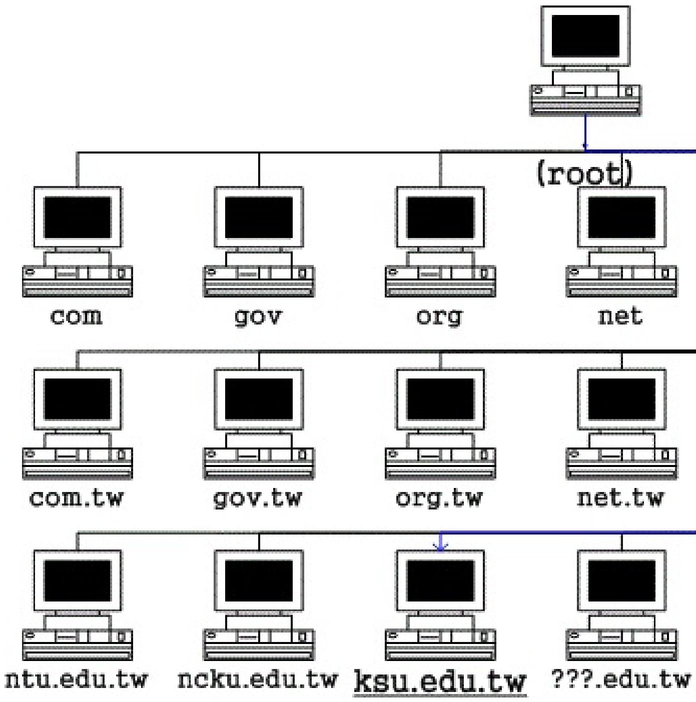


图 19.1-3、从最上层到昆山科大之间的 DNS 阶层示意图

在整个 DNS 系统的最上方一定是 . (小数点) 这个 DNS 服务器 (称为 root)，最早以前它底下管理的就只有 (1)com, edu, gov, mil, org, .net 这种特殊领域以及 (2)以国家为分类的第二层的主机名了！这两者称为 Top Level Domains (TLDs) 喔！

- 一般最上层领域名 (Generic TLDs, gTLD)：例如 .com, .org, .gov 等等
- 国码最上层领域名 (Country code TLDs, ccTLD)：例如 .tw, .uk, .jp, .cn 等

先来谈谈一般最上层领域 (gTLD) 好了，最早 root 仅管理六大领域名，分别如下：

名称	代表意义
com	公司、行号、企业
org	组织、机构
edu	教育单位
gov	政府单位
net	网络、通讯
mil	军事单位

但是因特网成长的速度太快了，因此后来除了上述的六大类别之外，还有诸如 .asia, .info, .jobs (注1) 等领域名的开放。此外，为了让某些国家也能够有自己的最上层领域名，因此，就有所谓的 ccTLD 了。这样做有什么好处呢？因为自己的国家内有最上层 ccTLD ，所以如果有 domain name 的需求，则只要向自己的国家申请即可，不需要再到最上层去申请啰！

- 授权与分层负责

既然 TLD 这么好，那么是否我们可以自己设定 TLD 呢？当然不行！因为我们得向上层 ISP 申请领域名的授权才行。例如台湾地区最上层的领域名是以 .tw 为开头，管理这个领域名的机器 IP 是在台湾，但是 .tw 这部服务器必须向 root (.) 注册领域名查询授权才行 (如上图 19.1-3 所示)。

那么每个国家之下记录的主要下层有哪些领域呢？基本上就是原先 root 管理的那六大类。 不过，由于各层 DNS 都能管理自己辖下的主机名或子领域，因此，我们的 .tw 可以自行规划自己的子领域名喔！ 例如目前台湾 ISP 常提供的 .idv.tw 的个人网站就是一例啊！

再强调一次，DNS 系统是以所谓的阶层式的管理，所以，请注意喔！那个 .tw 只记录底下那一层的这数个主要的 domain 的主机而已！至于例如 edu.tw 底下还有个 ksu.edu.tw 这部机器，那就直接授权交给 edu.tw 那部机器去管理了！也就是说『 每个上一层的 DNS 服务器所记录的信息，其实只有其下一层的主机名而已！ 』至于再下一层，则直接『授权』给下层的某部主机来管理啰！呵呵！所以你就应该会知道 DNS 到底是如何管理的吧！

会这样设定的原因不是没有道理的！这样设计的好处就是：每部机器管理的只有下一层的 hostname 对应 IP 而已，所以减少了管理上的困扰！而下层 Client 端如果有问题，只要询问上一层的 DNS server 即可！不需要跨越上层，除错上面也会比较简单呢！

- 透过 DNS 查询主机名 IP 的流程

刚刚说过 DNS 是以类似『树状目录』的型态来进行主机名的管理的！所以每一部 DNS 服务器都『仅管理自己的下一层主机名的转译』而已， 至于下层的下层，则『授权』给下层的 DNS 主机来管理啦！这样说好像很绕口，好吧！我们就以下图来说一说原理啰：

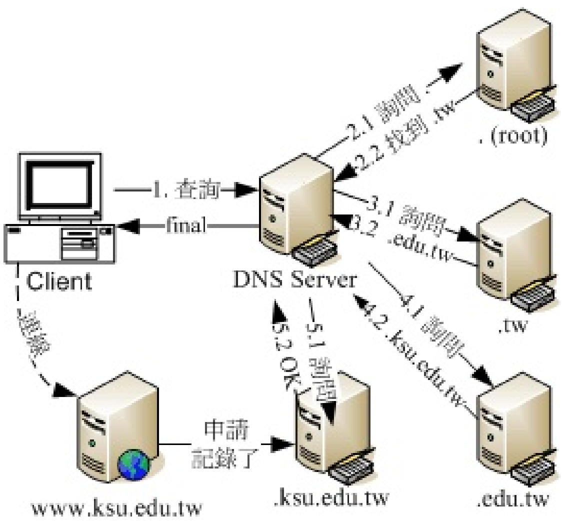


图 19.1-4、透过 DNS 系统查询主机名解译的流程

首先，当你在浏览器的网址列输入 http://www.ksu.edu.tw 时，你的计算机就会依据相关设定 (在 Linux 底下就是利用 /etc/resolv.conf 这个档案) 所提供的 DNS 的 IP 去进行联机查询了。由于目前最常见的 DNS 服务器就属 Hinet 的 168.95.1.1 这个 DNS，所以我们就拿他来做例子吧！嗯！这个时候，hinet 的这部服务器会这样工作：

1. 收到用户的查询要求，先查看本身有没有纪录，若无则向 . 查询： 由于 DNS 是阶层式的架构，每部主机都会管理自己辖下的主机名解译而已。因为 hinet 并没有管理台湾学术网络的权力， 因此就无法直接回报给客户端。此时 168.95.1.1 就会向最顶层，也就是 . (root) 的服务器查询相关 IP 信息。
2. 向最顶层的 . (root) 查询： 168.95.1.1 会主动的向 . 询问 www.ksu.edu.tw 在哪里呢？但是由于 . 只记录了 .tw 的信息 (因为台湾只有 .tw 向 . 注册而已)，此时 . 会告知『我是不知道这部主机的 IP 啦，不过，你应该向 .tw 去询问才对，我这里不管！ 我跟你说 .tw 在哪里吧！』
3. 向第二层的 .tw 服务器查询： 168.95.1.1 接着又到 .tw 去查询，而该部机器管理的又仅有 .edu.tw, .com.tw, gov.tw... 那几部主机，经过比对后发现我们要的是 .edu.tw 的网域，所以这个时候 .tw 又告诉 168.95.1.1 说：『你要去管理 .edu.tw 这个网域的主机那里查询，我有他的 IP ！』
4. 向第三层的 .edu.tw 服务器查询： 同理可证， .edu.tw 只会告诉 168.95.1.1 ，应该要去 .ksu.edu.tw 进行查询，这里只能告知 .ksu.edu.tw 的 IP 而已。
5. 向第四层的 .ksu.edu.tw 服务器查询： 等到 168.95.1.1 找到 .ksu.edu.tw 之后， Bingo ！.ksu.edu.tw 说：『没错！这部主机名是我管理的～ 我跟你说他的 IP 是...所以此时 168.95.1.1 就能够查到 www.ksu.edu.tw 的 IP 啰！
6. 记录暂存内存并回报用户： 查到了正确的 IP 后，168.95.1.1 的 DNS 机器总不会在下次有人查询 www.ksu.edu.tw 的时候再跑一次这样的流程吧！ 粉远粉累的呐！而且也很耗系统的资源与网络的带宽，所以呢，168.95.1.1 这个 DNS 会很聪明的先记录一份查询的结果在自己的暂存内存当中，以方便响应下一次的相同要求啊！ 最后则将结果回报给 client 端！当然啦，那个记忆在 cache 当中的数据，其实是有时间性的，当过了 DNS 设定记忆的时间 (通常可能是 24 小时)，那么该记录就会被释放喔！

整个分层查询的流程就是这样，总是得要先经过 . 来向下一层进行查询，最终总是能得到答案的。这样分层的好处是：

- 主机名修改的仅需自己的 DNS 更动即可，不需通知其他人： 当一个『合法』的 DNS 服务器里面的设定修改了之后，来自世界各地任何一个 DNS 的要求，都会正确无误的显示正确的主机名对应 IP 的信息，因为他们会一层一层的寻找下来。所以，要找你的主机名对应的 IP 就一定得要透过你的上层 DNS 服务器的纪录才行！因此，只要你的主机名字是经过上层『合法的 DNS』服务器设定的，那么就可以在 Internet 上面被查询到啦！呵呵！很简单维护吧，机动性也很高。
- DNS 服务器对主机名解析结果的快取时间： 由于每次查询到的结果都会储存在 DNS 服务器的高速缓存中，以方便若下次有相同需求的解析时，能够快速的响应。 不过，查询结果已经被快取了，但是原始 DNS 的主机名与 IP 对应却修改了，此时若有人再次查

询， 系统可能会回报旧的 IP 喔！所以，在快取内的答案是有时间性的！通常是数十分钟到三天之内。 这也是为什么我们常说当你修改了一个 domain name 之后，可能要 2 ~ 3 天后才能全面的启用的缘故啦！

- 可持续向下授权 (子领域名授权)： 每一部可以记录主机名与 IP 对应的 DNS 服务器都可以随意更动他自己的数据库对应， 因此主机名与域名在各个主机底下都不相同。举例来说， idv.tw 是仅有台湾才有这个 idv 的网域～ 因为这个 idv 是由 .tw 所管理的，所以只要台湾 .tw 维护小组同意，就能够建立该网域喔！

好啦！既然 DNS 这么棒，然后我们又需要架站，所以需要一个主机的名称，那么我们需要架设 DNS 了吗？当然不是，为什么呢？刚刚鸟哥提到了很多次的『合法』的字眼，因为他就牵涉到『授权』的问题了！ 我们在第十章当中也提到，只要主机名合法即可，不见得需要架设 DNS 的啦！

例题：透过 dig 实作出本小节谈到的 . --> .tw --> .edu.tw --> .ksu.edu.tw --> www.ksu.edu.tw 的查询流程，并分析每个查询阶段的 DNS 服务器有几部？答：事实上，我们可以透过第四章约略谈过的 dig 这个指令来实作出喔！使用追踪功能 (+trace) 就能够达到这个目的了。使用方式如下：

```
[root@www ~]# dig +trace www.ksu.edu.tw
; &lt;&lt;&gt;&gt; DiG 9.3.6-P1-RedHat-9.3.6-16.P1.el5 &lt;&lt;&gt;&gt;+trace www.ksu.edu
;; global options:  printcmd
.                        486278  IN      NS      a.root-servers.net.
.                        486278  IN      NS      b.root-servers.net.
....(底下省略)....
# 上面的部分在追踪 . 的服务器，可从 a ~ m.root-servers.net.
;; Received 500 bytes from 168.95.1.1#53(168.95.1.1) in 22 ms

tw.                      172800  IN      NS      ns.twnic.net.
tw.                      172800  IN      NS      a.dns.tw.
tw.                      172800  IN      NS      b.dns.tw.
....(底下省略)....
# 上面的部分在追踪 .tw. 的服务器，可从 a ~ h.dns.tw. 包括 ns.twnic.net.
;; Received 474 bytes from 192.33.4.12#53(c.root-servers.net) in 168 ms

edu.tw.                  86400   IN      NS      a.twnic.net.tw.
edu.tw.                  86400   IN      NS      b.twnic.net.tw.
# 追踪 .edu.tw. 的则有 7 部服务器
;; Received 395 bytes from 192.83.166.11#53(ns.twnic.net) in 22 ms

ksu.edu.tw.              86400   IN      NS      dns2.ksu.edu.tw.
ksu.edu.tw.              86400   IN      NS      dns3.twaren.net.
ksu.edu.tw.              86400   IN      NS      dns1.ksu.edu.tw.
;; Received 131 bytes from 192.83.166.9#53(a.twnic.net.tw) in 22 ms

www.ksu.edu.tw.          3600    IN      A       120.114.100.101
ksu.edu.tw.              3600    IN      NS      dns2.ksu.edu.tw.
ksu.edu.tw.              3600    IN      NS      dns1.ksu.edu.tw.
ksu.edu.tw.              3600    IN      NS      dns3.twaren.net.
;; Received 147 bytes from 120.114.150.1#53(dns2.ksu.edu.tw) in 14 ms
```

最终的结果有找到 A (Address) 是 120.114.100.101，不过这个例题的重点是，要让大家瞧瞧整个 DNS 的搜寻过程！ 在 dig 加上 +trace 的选项后，就能够达到这个目的。至于其他的都是服务器 (NS) 的设定值与追踪过程喔！ 有没有很清楚啊？^_^。至于 A 与 NS 等相关的数

据，我们在后续的 DNS 数据库介绍中，再分别介绍啰。

- DNS 使用的 port number

好了，既然 DNS 系统使用的是网络的查询，那么自然需要有监听的 port 啰！没错！很合理！那么 DNS 使用的是那一个 port 呢？那就是 53 这个 port 啦！你可以到你的 Linux 底下的 /etc/services 这个档案看看！搜寻一下 domain 这个关键词，就可以查到 53 这个 port 啦！

但是这里需要跟大家报告的是，通常 DNS 查询的时候，是以 udp 这个较快速的数据传输协议来查询的， 但是万一没有办法查询到完整的信息时，就会再次的以 tcp 这个协定来重新查询的！所以启动 DNS 的 daemon (就是 named 啦) 时，会同时启动 tcp 及 udp 的 port 53 喔！所以，记得防火墙也要同时放行 tcp, udp port 53 呢！

19.1.3 合法 DNS 的关键：申请领域查询授权

什么？DNS 服务器的架设还有『合法』与『不合法』之分喔？不是像其他的服务器一样，架设好之后人家就查的到吗？ 非也非也！为什么呢？底下我们就来谈一谈。

- 向上层领域注册取得合法的领域查询授权

我们在第十章也讲过，申请一个合法的主机名就是需要注册， 注册就是需要花钱啦！那么注册取得的资料有两种，一种是第十章谈到的 FQDN (主机名)，一种就是申请领域查询权。所谓的 FQDN 就是我们只需要主机名，详细的设定数据就由 ISP 帮我们搞定。例如图 19.1-4 所示， 那部 www.ksu.edu.tw 的详细主机名对应 IP 的数据就是请管理 .ksu.edu.tw 那个领域的服务器搞定的。

那什么是领域查询授权呢？同样用图 19.1-4 来解释，我们的 .ksu.edu.tw 必须要向 .edu.tw 那部主机注册申请领域授权，因此，未来有任何 .ksu.edu.tw 的要求时， .edu.tw 都会说：『我不知道！ 详情请去找 .ksu.edu.tw 吧！』此时，我们就得要架设 DNS 服务器来设定 .ksu.edu.tw 相关的主机名对应才行喔！ 是否很像人类社会的『授权』的概念？

也就是说，当你老板充分的『授权』给你某项工作的时候，从此，要进行该项工作的任何人， 从老板那边知道你才是真正『有权』的人之后，都必须要向你请示一样！^_^！所以啰，如果你要架设 DNS ，而且是可以连上 Internet 上面的 DNS 时，你就必须要透过『上层 DNS 服务器的授权』才行！这是很重要的观念喔！

让我们归纳一下，要让你的主机名对应 IP 且让其他计算机都可以查询的到，你有两种方式：

1. 上层 DNS 授权领域查询权，让你自己设定 DNS 服务器，或者是；
2. 直接请上层 DNS 服务器来帮你设定主机名对应！

3. 拥有领域查询权后，所有的主机名信息都以自己为准，与上层无关

很多朋友可能都有过申请 DNS 领域查询授权的经验，在申请时，ISP 就会要你填写 (1)你的 DNS 服务器名称以及 (2)该服务器的 IP。既然已经在 ISP 就填写了主机名与 IP 的对应，所以，即使我的 DNS 服务器挂点了，在 ISP 上面的主机名应该还是查到的 IP 吧？答案是：『错！』查不到的！为什么呢？

DNS 系统记录的信息非常的多，不过重点其实有两个，一个是记录服务器所在的 NS (NameServer) 标志，另一个则是记录主机名对应的 A (Address) 标志。我们在网络上面查询到的最终结果，都是查询 IP (IP Address) 的，因此最终的标志要找的是 A 这个记录才对！我们以鸟哥注册的 .vbird.org 来说明好了，鸟哥去注册时， 记录在 ISP 的 DNS 服务器名称为 dns.vbird.org，该笔记录其实就是 NS ，并非 A ，如下图所示：

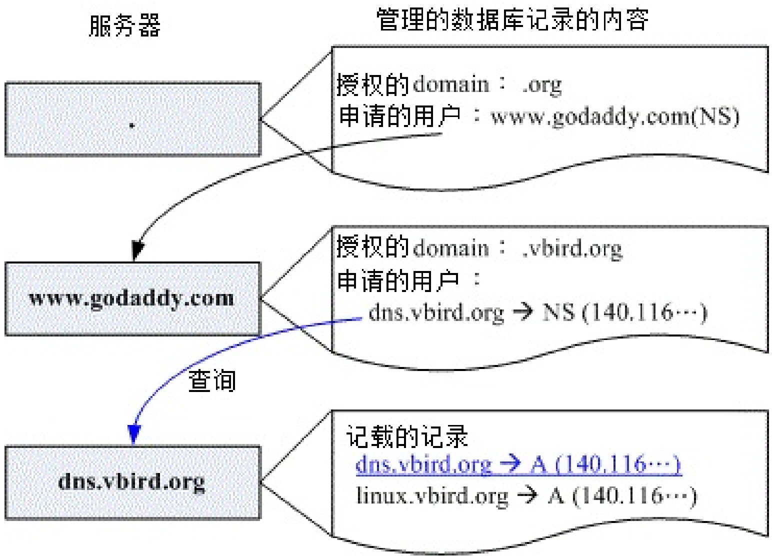


图 19.1-5、记录的授权主机名与实际 A 记录的差异

上图中，虽然在 godaddy 服务器内有记录一笔『要查询 .vbird.org 时，请到 dns.vbird.org (NS) 去查，这个管理者的 IP 是 140.116...』，但是这笔记录只是告诉我们要去下一个服务器找，并不是最终的 A (IP Address) 的答案，所以还得要继续往下找 (随时记得图 19.1-4 的查询流程)。此时，有几种结果会导致 dns.vbird.org 的 IP 找不到，或者是最终的 IP 与 godaddy 记录的不同的结果喔！那就是：

- dns.vbird.org 服务器挂点时： 如果 dns.vbird.org 这部主机挂点，那么在上图显示『查询』箭头的步骤会被中断，因此就会出现『联机不到 dns.vbird.org 的 IP』的结果。因为无论如何，DNS 系统都会去找到最后一个含有 A 地址的记录啊！
- dns.vbird.org 服务器内的数据库忘记补上数据时： 如果鸟哥在自己的服务器数据库中，忘记加上 dns.vbird.org 的记录时，最终的结果还是会显示『找不到该服务器的 IP』；
- dns.vbird.org 服务器内的数据库数据编写不一致时： 如果是在鸟哥自己服务器的数据库内的 dns.vbird.org 所记录的 IP 与 godaddy 的不同，最终的结果会以鸟哥记录的为准。

总之，你在 ISP 上面填写的主机名只是一个参考用的，最终还是要在你自己 DNS 服务器当中设定好才行！ 虽然可以自己恶搞一下，不过，通常大家还是会让 ISP 上面的 DNS 服务器主机名与自己的数据库主机名一致， 亦即上图中，中间与最下面方框内的 dns.vbird.org 的 NS 及 A 都对应到同一个 IP 就是了。

19.1.4 主机名交由 ISP 代管还是自己设定 DNS 服务器

前面 19.1.3 小节以及第十章都谈过，申请主机名或域名主要有两种方式，就是刚刚上头提到的 DNS 授权，或者是直接交给 ISP 来管理。交给 ISP 管理的，就可以称作是域名代管啦！当然啦，如果你是学校单位的话， 或者是企业内部的小单位，那么就得请你向上层 DNS 主机的负责人要求啰！无论如何，你只能有两个选择就是了，要不就是请他帮忙你设定好 hostname 对应 IP ，要嘛就是请他直接将某个 domain name 段授权给你做为 DNS 的主要管理网域。

那么我怎么知道那个方式对我比较好呢？请注意，由于 DNS 架设之后，会多出一个监听的 port ，所以理论上，是比较不安全的！而且，由于因特网现在都是透过主机名在联机，在了解上面谈到的主机名查询流程后， 你会发现，DNS 设定错误是很要命的！因为你的主机名再也找不到了。所以，这里的建议是：

- 需要架设 DNS 的时机：
- 你所负责需要连上 Internet 的主机数量庞大：例如你一个人负责整个公司十几部的网络 Server，而这些 Server 都是挂载你的公司网域之下的。这个时候想要不架设 DNS 也粉难啦！
- 你可能需要时常修改你 Server 的名字，或者是你的 Server 有随时增加的可能性与变动性；
- 不需要架设 DNS 的时机：
- 网络主机数量很少：例如家里或公司只有需要一部 mail server 时；
- 你可以直接请上层 DNS 主机管理员帮你设定好 Hostname 的对应时；
- 你对于 DNS 的认知不足时，如果架设反而容易造成网络不通的情况；
- 架设 DNS 的费用很高时！

19.1.5 DNS 数据库的记录：正解, 反解, Zone 的意义

从前面的图 19.1-4 的查询流程中，我们知道最重要的就是 .ksu.edu.tw 那部 DNS 服务器内的记录信息了。这些记录的咚咚我们可以称呼为数据库，而在数据库里面针对每个要解析的领域 (domain)，就称为一个区域 (zone)。那么到底有哪些要解析的领域呢？基本上，有从主机

名查到 IP 的流程，也可以从 IP 反查到主机名的方式。 因为最早前 DNS 的任务就是要将主机名解析为 IP，因此：

- 从主机名查询到 IP 的流程称为：正解
- 从 IP 反解析到主机名的流程称为：反解
- 不管是正解还是反解，每个领域的记录就是一个区域 (zone)

举例来说，昆山科大 DNS 服务器管理的就是 *.ksu.edu.tw* 这个领域的查询权，任何想要知道 .ksu.edu.tw 主机名的 IP 都得向昆山科大的 DNS 服务器查询，此时 .ksu.edu.tw 就是一个『正解的领域』。而昆山科大有申请到几个 class C 的子域， 例如 120.114.140.0/24，如果这 254 个可用 IP 都要设定主机名，那么这个 120.114.140.0/24 就是一个『反解的领域』！ 另外，每一部 DNS 服务器都可以管理多个领域，不管是正解还是反解。

- 正解的设定权以及 DNS 正解 zone 记录的标志

那谁可以申请正解的 DNS 服务器架设权呢？答案是：都可以！只要该领域没有人使用， 那你先抢到了，就能够使用了。不过，因为国际 INTERNIC 已经定义出 gTLD 以及 ccTLD 了，所以你不能自定义例如 centos.vbird 这种网域的！还是得要符合上层 DNS 所给予的领域范围才行。举例来说，台湾个人网站就常使用 *.idv.tw 这样的领域名。

那正解文件的 zone 里面主要记录了什么东西呢？因为正解的重点在由主机名查询到 IP，而且每部 DNS 服务器还是得要定义清楚，同时，你可能还需要架设 master/slave 架构的 DNS 环境，因此，正解 zone 通常具有底下几种标志：

- SOA：就是开始验证 (Start of Authority) 的缩写，相关资料本章后续小节说明；
- NS：就是名称服务器 (NameServer) 的缩写，后面记录的数据是 DNS 服务器的意思；
- A：就是地址 (Address) 的缩写，后面记录的是 IP 的对应 (最重要)；

- 反解的设定权以及 DNS 反解 zone 记录的标志

正解的领域名只要符合 INTERNIC 及你的 ISP 规范即可，取得授权较为简单 (自己取名字)。那反解呢？反解主要是由 IP 找到主机名，因此重点是 IP 的所有人是谁啦！因为 IP 都是 INTERNIC 发放给各家 ISP 的，而且我们也知道，IP 可不能乱设定 (路由问题)！所以啰，能够设定反解的就只有 IP 的拥有人，亦即你的 ISP 才有权力设定反解的。那你向 ISP 取得的 IP 能不能自己设定反解呢？答案是不行！除非你取得的是整个 class C 以上等级的 IP 网段，那你的 ISP 才有可能给你 IP 反解授权。否则，若有反解的需求，就得要向你的直属上层 ISP 申请才行！

那么反解的 zone 主要记录的信息有哪些呢？除了服务器必备的 NS 以及 SOA 之外，最重要的就是：

- PTR：就是指向 (PoinTeR) 的缩写，后面记录的数据就是反解到主机名啰！

- 每部 DNS 都需要的正解 zone： hint

现在你知道一个正解或一个反解就可以称为一个 zone 了！那么有没有那个 zone 是特别重要的呢？有的，那就是 . 啊！ 从图 19.1-4 里面我们就知道，当 DNS 服务器在自己的数据库找不到所需的信息时， 一定会去找 . ，那 . 在哪里啊？所以就得要有记录 . 在哪里的记录 zone 才行啊！这个记录 . 的 zone 的类型，就被我们称为 hint 类型！这几乎是每个 DNS 服务器都得要知道的 zone 喔！

所以说，一部简单的正解 DNS 服务器，基本上就要有两个 zone 才行，一个是 hint ，一个是关于自己领域的正解 zone。举鸟哥注册的 vbird.org 为例，在鸟哥的 DNS 服务器内，至少就要有这两个 zone：

- hint (root)：记录 . 的 zone；
- vbird.org：记录 .vbird.org 这个正解的 zone。

你会发现我没有 vbird.org 这个 domain 所属 IP 的反解 zone ，为什么呢？请参考上面的详细说明吧！ 简单的说，就是因为反解需要要求 IP 协议的上层来设定才行！

- 正反解是否一定要成对？

好了，正反解需不需要成套产生，在这里不用多说明了吧？^^！请注意喔，在很多的情况下，尤其是目前好多莫名其妙的领域名产生出来，所以，常常会只有正解的设定需求而已。不过也不需要太过担心啦， 因为通常在反查的情况中，如果你是使用目前台湾地区最流行的 *ADSL* 上网的话，那么 *ISP* 早就已经帮你设定好反解了！例如：*211.74.253.91* 这个 *seednet* 的浮动式 *IP* 反查的结果会得到 *211-74-253-91.adsl.dynamic.seed.net.tw.* 这样的主机名！所以在一般我们自行申请领域名的时候，你只要担心正解的设定即可！ 不然的话，反正反解的授权根本也不会开放给你，你自己设定得很高兴也没有用呀！ ^^

事实上，需要正反解成对需求的大概仅有 mail server 才需要吧！由于目前网络带宽老是被垃圾、广告邮件占光， 所以 Internet 的社会对于合法的 mail server 规定也就越来越严格。如果你想要架设 mail server 时， 最好具有固定 IP ，这样才能向你的 ISP 要求设定反解喔！以 hinet 为例的反解申请：

- http://hidomain.hinet.net/top1.html

19.1.6 DNS 数据库的类型：hint, master/slave 架构

你知道的，DNS 越来越重要，所以，如果你有注册过领域名的话，就可以发现，现在 ISP 都要你填写两部 DNS 服务器的 IP 哩！因为要作为备援之用嘛！总不能一部 DNS 挂点后，害你的所有主机名都不能被找到～那真麻烦～

但是，如果有两部以上的 DNS 服务器，那么网络上会搜寻到哪一部呢？答案是，不知道！因为是随机的～ 所以，如果你的领域有两部 DNS 服务器的话，那这两部 DNS 服务器的内容就得完全一模一样，否则，由于是随机找到 DNS 来询问，因此若数据不同步，很可能造成其他用户无法取得正确数据的问题。

为了解决这个问题，因此在 . (root) 这个 hint 类型的数据库档案外，还有两种基本类型，分别是 Master (主人、主要) 数据库与 Slave (奴隶、次要) 数据库类型。这个 Master/Slave 就是要用来解决不同 DNS 服务器上面的数据同步问题的。 所以底下让我们来聊聊 Master/Slave 吧！

- Master：

这种类型的 DNS 数据库中，里面所有的主机名相关信息等，通通要管理员自己手动去修改与设定， 设定完毕还得要重新启动 DNS 服务去读取正确的数据库内容，才算完成数据库更新。一般来说，我们说的 DNS 架设，就是指设定这种数据库的类型。同时，这种类型的数据库，还能够提供数据库内容给 slave 的 DNS 服务器喔！

- Slave：

如前所述，通常你不会只有一部 DNS 服务器，例如我们前面的例题查询到的 .ksu.edu.tw 就有 3 部 DNS 服务器来管理自己的领域。那如果每部 DNS 我们都是使用 Master 数据库类型，当有用户向我要求要修改或者新增、删除数据时， 一笔数据我就得要做三次，还可能会不小心手滑导致某几部出现错误，此时可就伤脑筋了～因此，这时使用 Slave 类型的数据库取得方式就很有用！

Slave 必须要与 Master 相互搭配，若以 .ksu.edu.tw 的例子来说，如果我必须要有三部主机提供 DNS 服务，且三部内容相同， 那么我只要指定一部服务器为 Master ，其他两部为该 Master 的 Slave 服务器，那么当要修改一笔名称对应时，我只要手动更改 Master 那部机器的配置文件，然后，重新启动 BIND 这个服务后，呵呵！其他两部 Slave 就会自动的被通知更新了！这样一来，在维护上面可就轻松写意的多了～

Tips: 如果你设定 Master/Slave 架构时，你的 Master 主机必须要限制 只有某些特定 IP 的主机能够取得你 Master 主机的正反解数据库权限才好！ 所以，上面才会提到 Master/Slave 必须要互相搭配才行！



- Master / Slave 的查询优先权？

另外，既然我的所有 DNS 服务器是需要同时提供 internet 上面的领域名解析的服务， 所以不论是 Master 还是 Slave 服务器，他都必须要可以同时提供 DNS 的服务才好！ 因为在 DNS 系统当中，领域名的查询是『先抢先赢』的状态，我们不会晓得哪一部主机的数据会先被查询到的！ 为了提供良好的 DNS 服务，每部 DNS 主机都要能正常工作才好啊！而且，每一部 DNS 服务器的数据库内容需要完全一致，否则就会造成客户端找到的 IP 是错误的！

- Master / Slave 数据的同步化过程

那么 Master/Slave 的数据更新到底是如何动作的呢？请注意，Slave 是需要更新来自 Master 的数据啊！所以当然 Slave 在设定之初就需要存在 Master 才行喔！基本上，不论 Master 还是 Slave 的数据库，都会有一个代表该数据库新旧的『序号』，这个序号数值的大小，是会影响是否要更新的动作唷！ 至于更新的方式主要有两种：

- Master 主动告知：例如在 Master 在修改了数据库内容，并且加大数据库序号后， 重新启动 DNS 服务，那 master 会主动告知 slave 来更新数据库，此时就能够达成数据同步；
- 由 Slave 主动提出要求：基本上， Slave 会定时的向 Master 察看数据库的序号， 当发现 Master 数据库的序号比 Slave 自己的序号还要大 (代表比较新)，那么 Slave 就会开始更新。如果序号不变， 那么就判断数据库没有更动，因此不会进行同步更新。

由上面的说明来看，其实设计数据库的序号最重要的目的就是让 master/slave 数据的同步化。那我们也知道 slave 会向 master 提出数据库更新的需求，问题是，多久提出一次更新，如果该次更新时由于网络问题，所以没有查询到 master 的序号 (亦即更新失败)，那隔多久会重新更新一次？这个与 SOA 的标志有关，后续谈到正、反解数据库后， 再来详细说明吧！

如果你想要架设 Master/Slave 的 DNS 架构时，两部主机 (Master/Slave) 都需要你能够掌控才行！网络上很多的文件在这个地方都有点『闪失』，请特别的留意啊！因为鸟哥的 DNS 服务器常常会听到某些其他 DNS 的数据库同步化需求，眞觉得烦呐！

19.2 Client 端的设定

由于 DNS 是每部想要连上因特网的主机都得要设定的，因此我们就从简单的客户端设定谈起。因为未来架设好 DNS server 后，我们都会直接进行测试，所以，这个部分得先处理处理比较妥当啊！

19.2.1 相关配置文件

从 19.1.1 的说明当中我们晓得主机名对应到 IP 有两种方法，早期的方法是直接写在档案里面来对应， 后来比较新的方法则是透过 DNS 架构！那么这两种方法分别使用什么配置文件？可不可以同时存在？ 若同时存在时，那个方法优先？嗯！我们先来谈一谈几个配置文件吧！

- /etc/hosts ：这个是最早的 hostname 对应 IP 的档案；
- /etc/resolv.conf ：这个重要！就是 ISP 的 DNS 服务器 IP 记录处；
- /etc/nsswitch.conf：这个档案则是在『决定』先要使用 /etc/hosts 还是 /etc/resolv.conf 的设定！

一般而言， Linux 的预设主机名与 IP 的对应搜寻都以 /etc/hosts 为优先，为什么呢？你可以查看一下 /etc/nsswitch.conf ，并找到 hosts 的项目：

```
[root@www ~]# vim /etc/nsswitch.conf
hosts:      files dns
```

上面那个 files 就是使用 /etc/hosts，而最后的 dns 则是使用 /etc/resolv.conf 的 DNS 服务器来进行搜寻啦！因此，你可以先以 /etc/hosts 来设定 IP 对应！当然啦，你也可以将他调换过来，不过，总是 /etc/hosts 比较简单，所以将他摆在前面比较好啦！

好啦，既然我们是要进行 DNS 测试的，那么就得要了解一下 /etc/resolv.conf 的内容，假设你在台湾，使用的是 hinet 的 168.95.1.1 这部 DNS 服务器，所以你应该这样写：

```
[root@www ~]# vim /etc/resolv.conf
nameserver 168.95.1.1
nameserver 139.175.10.20
```

DNS 服务器的 IP 可以设定多个，为什么要设定多个呢？因为当第一部 (照设定的顺序) DNS 挂点时，我们客户端可以使用第二部 (上述是 139.175.10.20) 来进行查询，这多少有点像 DNS 备援功能。 通常建议至少填写两部 DNS 服务器的 IP，不过在网络正常使用的情况下，永远只有第一部 DNS 服务器会被使用来查询，其他的设定值只是在第一部出问题时才会被使用。

Tips: 尽量不要设定超过 3 部以上的 DNS IP 在 /etc/resolv.conf 中，因为如果是你的区网出问题，导致无法联机到 DNS 服务器，那么你的主机还是会向每部 DNS 服务器发出联机要求，每次联机都有 timeout 时间的等待，会导致浪费非常多的时间喔！



例题：我的主机使用 DHCP 取得 IP ，很奇怪的，当我修改过 /etc/resolv.conf 之后，隔不多久这个档案又会恢复成原本的样子，这是什么原因？ 该如何处理？答：因为使用 DHCP 时，系统会主动的使用 DHCP 服务器传来的数据进行系统配置文件的修订。因此，你必须告知系统，不要使用 DHCP 传来的服务器设定值。 此时，你得要在 /etc/sysconfig/network-scripts/ifcfg-eth0 等相关档案内，增加一行：『PEERDNS=no』，然后重新启动网络即可。

此外，如果你有启动 CentOS 6.x 的 NetworkManager 服务，有时候也可能会产生一些奇特的现象哩！所以鸟哥是建议关掉它的！^_^

19.2.2 DNS 的正、反解查询指令： host, nslookup, dig

测试 DNS 的程序有很多，我们先来使用最简单的 host 吧！然后还有 nslookup 及 dig 哩！

- host

```
[root@www ~]# host [-a] FQDN [server]
[root@www ~]# host -l domain [server]
选项与参数：
-a ：代表列出该主机所有的相关信息，包括 IP、TTL 与除错讯息等等
-l ：若后面接的那个 domain 设定允许 allow-transfer 时，则列出该 domain
     所管理的所有主机名对应数据！
server：这个参数可有可无，当想要利用非 /etc/resolv.conf 内的 DNS 主机
        来查询主机名与 IP 的对应时，就可以利用这个参数了！

# 1\. 使用默认值来查出 linux.vbird.org 的 IP
[root@www ~]# host linux.vbird.org
linux.vbird.org has address 140.116.44.180             &lt;==这是 IP
linux.vbird.org mail is handled by 10 linux.vbird.org. &lt;==这是 MX (后续章节说明)

# 2\. 查出 linux.vbird.org 的所有重要参数
[root@www ~]# host -a linux.vbird.org
Trying "linux.vbird.org"
;; -&gt;&gt;HEADER&lt;&lt;- opcode: QUERY, status: NOERROR, id: 56213
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 0

;; QUESTION SECTION:
;linux.vbird.org.               IN      ANY

;; ANSWER SECTION:
linux.vbird.org.        145     IN      A       140.116.44.180

;; AUTHORITY SECTION:
vbird.org.              145     IN      NS      dns.vbird.org.
vbird.org.              145     IN      NS      dns2.vbird.org.

Received 86 bytes from 168.95.1.1#53 in 15 ms  &lt;==果然是从 168.95.1.1 取得的资料
# 看样子，不就是 dig 的输出结果？所以，我们才会说，使用 dig 才是王道！

# 3\. 强制以 139.175.10.20 这部 DNS 主机来查询
[root@www ~]# host linux.vbird.org 139.175.10.20
Using domain server:
Name: 139.175.10.20
Address: 139.175.10.20#53
Aliases:

linux.vbird.org has address 140.116.44.180
linux.vbird.org mail is handled by 10 linux.vbird.org.
```

看到最后一个范例，有注意到上面输出的特殊字体部分吗？很多朋友在测试自己的 DNS 时，常常会『指定到错误的 DNS 查询主机』了～因为他们的 /etc/reslov.conf 忘记改，所以老是找不到自己设定的数据库 IP 数据。所以你要仔细看啊！

```
# 4\. 找出 vbird.org 领域的所有主机对应
[root@www ~]# host -l vbird.org
; Transfer failed.
Host vbird.org not found: 9(NOTAUTH)
; Transfer failed. &lt;==竟然失败了！请看底下的说明！
```

怎么会无法响应呢？这样的响应是因为管理 vbird.org 领域的 DNS 并不许我们的领域查询，毕竟我们不是 vbird.org 的系统管理员，当然没有权限可以读取整个 vbird.org 的领域设定啰！这个『 host -l 』是用在自己的 DNS 服务器上，本章稍后谈到服务器设定后，使用这个选项就能够读取相关的数据了。

- nslookup

```
[root@www ~]# nslookup [FQDN] [server]
[root@www ~]# nslookup
选项与参数：
1\. 可以直接在 nslookup 加上待查询的主机名或者是 IP ，[server] 可有可无；
2\. 如果在 nslookup 后面没有加上任何主机名或 IP ，那将进入 nslookup 的查询功能
   在 nslookup 的查询功能当中，可以输入其他参数来进行特殊查询，例如：
   set type=any ：列出所有的信息『正解方面配置文件』
   set type=mx  ：列出与 mx 相关的信息！

# 1\. 直接搜寻 mail.ksu.edu.tw 的 IP 信息
[root@www ~]# nslookup mail.ksu.edu.tw
Server:         168.95.1.1
Address:        168.95.1.1#53  &lt;==还是请特别注意 DNS 的 IP 是否正确！

Non-authoritative answer:
Name:   mail.ksu.edu.tw
Address: 120.114.100.20        &lt;==回报 IP 给你啰！
```

nslookup 可单纯的将 hostname 与 IP 对应列出而已，不过，还是会将查询的 DNS 主机的 IP 列出来的！ 如果想要知道更多详细的参数，那可以直接进入 nslookup 这个软件的操作画面中，如下范例：

```
[root@www ~]# nslookup  &lt;==进入 nslookup 查询画面
&gt; 120.114.100.20         &lt;==执行反解的查询
&gt; www.ksu.edu.tw         &lt;==执行正解的查询
# 上面这两个仅列出正反解的信息，没有啥了不起的地方啦！
&gt; set type=any           &lt;==变更查询，不是仅有 A，全部信息都列出来
&gt; www.ksu.edu.tw
Server:         168.95.1.1
Address:        168.95.1.1#53

Non-authoritative answer:
Name:   www.ksu.edu.tw
Address: 120.114.100.101  &lt;==这是答案

Authoritative answers can be found from: &lt;==这是相关授权 DNS 说明
ksu.edu.tw      nameserver = dns2.ksu.edu.tw.
ksu.edu.tw      nameserver = dns1.ksu.edu.tw.
dns1.ksu.edu.tw internet address = 120.114.50.1
dns2.ksu.edu.tw internet address = 120.114.150.1
&gt; exit &lt;==离开吧！皮卡丘
```

在上面的案例当中，请注意，如果你在 nslookup 的查询画面当中，输入 set type=any 或其他参数， 那么就无法再进行反解的查询了！这是因为 any 或者是 mx 等等的标志都是记录在正解 zone 当中的缘故！

- dig (未来的主流，请爱用他！)

```
[root@www ~]# dig [options] FQDN [@server]
选项与参数：
@server ：如果不以 /etc/resolv.conf 的设定来作为 DNS 查询，可在此填入其他的 IP
options：相关的参数很多，主要有 +trace, -t type 以及 -x 三者最常用
  +trace ：就是从 . 开始追踪，在 19.1.2 里面谈过了！回头瞧瞧去！
  -t type：查询的数据主要有 mx, ns, soa 等类型，相关类型 19.4 来介绍
  -x      ：查询反解信息，非常重要的项目！

# 1\. 使用默认值查询 linux.vbird.org 吧！
[root@www ~]# dig linux.vbird.org
; &lt;&lt;&gt;&gt; DiG 9.7.0-P2-RedHat-9.7.0-5.P2.el6_0.1 &lt;&lt;&gt;&gt; linux.vbird.or
;; global options: +cmd
;; Got answer:
;; -&gt;&gt;HEADER&lt;&lt;- opcode: QUERY, status: NOERROR, id: 37415
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 0

;; QUESTION SECTION:     &lt;==提出的问题的部分
;linux.vbird.org.               IN      A

;; ANSWER SECTION:       &lt;==主要的回答阶段
linux.vbird.org.        600     IN      A       140.116.44.180

;; AUTHORITY SECTION:    &lt;==其他与此次回答有关的部分
vbird.org.              600     IN      NS      dns.vbird.org.
vbird.org.              600     IN      NS      dns2.vbird.org.

;; Query time: 9 msec
;; SERVER: 168.95.1.1#53(168.95.1.1)
;; WHEN: Thu Aug  4 14:12:26 2011
;; MSG SIZE  rcvd: 86
```

在这个范例当中，我们可以看到整个显示出的讯息包括有几个部分：

- QUESTION(问题)：显示所要查询的内容，因为我们是查询 linux.vbird.org 的 IP，所以这里显示 A (Address)；
- ANSWER(回答)：依据刚刚的 QUESTION 去查询所得到的结果，答案就是回答 IP 啊！
- AUTHORITY(验证)：由这里我们可以知道 linux.vbird.org 是由哪部 DNS 服务器所提供的答案！ 结果是 dns.vbird.org 及 dns2.vbird.org 这两部主机管理的。另外，那个 600 是啥咚咚？图 19.1-4 提到过的流程，就是允许查询者能够保留这笔记录多久的意思 (快取)，在 linux.vbird.org 的设定中，预设可以保留 600 秒。

```
# 2\. 查询 linux.vbird.org 的 SOA 相关信息吧！
[root@www ~]# dig -t soa linux.vbird.org
; &lt;&lt;&gt;&gt; DiG 9.7.0-P2-RedHat-9.7.0-5.P2.el6_0.1 &lt;&lt;&gt;&gt; -t soa linux.v
;; global options: +cmd
;; Got answer:
;; -&gt;&gt;HEADER&lt;&lt;- opcode: QUERY, status: NOERROR, id: 57511
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;linux.vbird.org.               IN      SOA

;; AUTHORITY SECTION:
vbird.org.              600     IN      SOA     dns.vbird.org. root.dns.vbird.org.
 2007091402 28800 7200 720000 86400

;; Query time: 17 msec
;; SERVER: 168.95.1.1#53(168.95.1.1)
;; WHEN: Thu Aug  4 14:15:57 2011
;; MSG SIZE  rcvd: 78
```

由于 dig 的输出信息实在是太丰富了，又分成多个部分去进行回报，因此很适合作为 DNS 追踪回报的一个指令呢！ 你可以透过这个指令来了解一下你所设定的 DNS 数据库是否正确，并进行除错喔！ ^_^！此外，你也可以透过『 -t type 』 的功能去查询其他服务器的设定值，可以方便你进行设定 DNS 服务器时的参考喔！正解查询完毕，接下来玩一玩反解吧！

```
# 3\. 查询 120.114.100.20 的反解信息结果
[root@www ~]# dig -x 120.114.100.20
; &lt;&lt;&gt;&gt; DiG 9.7.0-P2-RedHat-9.7.0-5.P2.el6_0.1 &lt;&lt;&gt;&gt; -x 120.114.100
;; global options: +cmd
;; Got answer:
;; -&gt;&gt;HEADER&lt;&lt;- opcode: QUERY, status: NOERROR, id: 60337
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 3, ADDITIONAL: 3

;; QUESTION SECTION:
;20.100.114.120.in-addr.arpa.   IN      PTR

;; ANSWER SECTION:
20.100.114.120.in-addr.arpa. 3600 IN    PTR     mail-out-r2.ksu.edu.tw.
20.100.114.120.in-addr.arpa. 3600 IN    PTR     mail-smtp-proxy.ksu.edu.tw.
20.100.114.120.in-addr.arpa. 3600 IN    PTR     mail.ksu.edu.tw.

;; AUTHORITY SECTION:
100.114.120.in-addr.arpa. 3600  IN      NS      dns1.ksu.edu.tw.
100.114.120.in-addr.arpa. 3600  IN      NS      dns3.twaren.net.
100.114.120.in-addr.arpa. 3600  IN      NS      dns2.ksu.edu.tw.

;; ADDITIONAL SECTION:
dns1.ksu.edu.tw.        3036    IN      A       120.114.50.1
dns2.ksu.edu.tw.        2658    IN      A       120.114.150.1
dns3.twaren.net.        449     IN      A       211.79.61.47

;; Query time: 29 msec
;; SERVER: 168.95.1.1#53(168.95.1.1)
;; WHEN: Thu Aug  4 14:17:58 2011
;; MSG SIZE  rcvd: 245
```

反解相当有趣！从上面的输出结果来看，反解的查询目标竟然从 120.114.100.20 变成了 20.100.114.120.in-addr.arpa. 这个模样～这是啥鬼东西？不要怕，这等我们讲到反解时再跟大家进一步解释。 你现在要知道的是，反解的查询领域名，跟正解不太一样即可，尤其是那

个怪异的 in-addr.arpa. 结尾的数据，可以先记下来。

19.2.3 查询领域管理者相关信息： whois

上个小节谈到的是主机名的正反解查询指令，那如果你想要知道整个领域的设定，使用的是『 host -l 领域名 』去查， 那如果你想要知道的是『这个领域是谁管的』的信息呢？那就得要使用 whois 这个指令才行喔！在 CentOS 6.x 当中， whois 是由 jwhois 这个软件提供的，因此，如果找不到 whois 时，请用 yum 去安装这个软件吧！

- whois

```
[root@www ~]# whois [domainname]  &lt;==注意啊！是 domain 而不是 hostname
[root@www ~]# whois centos.org
[Querying whois.publicinterestregistry.net]
[whois.publicinterestregistry.net]
# 这中间是一堆 whois 服务器提供的讯息告知！底下是实际注册的数据
Domain ID:D103409469-LROR
Domain Name:CENTOS.ORG
Created On:04-Dec-2003 12:28:30 UTC
Last Updated On:05-Dec-2010 01:23:25 UTC
Expiration Date:04-Dec-2011 12:28:30 UTC  &lt;==记载了建立与与失效的日期
Sponsoring Registrar:Key-Systems GmbH (R51-LROR)
Status:CLIENT TRANSFER PROHIBITED
Registrant ID:P-8686062
Registrant Name:CentOS Domain Administrator
Registrant Organization:The CentOS Project
Registrant Street1:Mechelsesteenweg 170
# 底下则是一堆联络方式，鸟哥将它取消了，免得多占篇幅～
```

whois 这个指令可以查询到当初注册这个 domain 的用户的相关信息。不过，由于近年来很多网络信息安全的问题，这个 whois 所提供的信息真的是太详细了，为了保护使用者的隐私权，所以，目前这个 whois 所查询到的信息已经不见得是完全正确的了～而且，在显示出 whois 的信息之前，还会有一段宣告事项的告知呢～ ^_^y

如果使用 whois 来检查鸟哥所注册的合法 domain 会是如何呢？看看：

```
[root@www ~]# whois vbird.idv.tw
[Querying whois.twnic.net]
[whois.twnic.net]           &lt;==这个 whois 服务器查到的数据
Domain Name: vbird.idv.tw   &lt;==这个 domain 的信息

   Contact:                 &lt;==联络者的联络方式
      Der-Min Tsai
      vbird@pc510.ev.ncku.edu.tw

   Record expires on 2018-09-17 (YYYY-MM-DD)
   Record created on 2002-09-13 (YYYY-MM-DD)

Registration Service Provider: HINET
```

呵呵！这个 domain 会在 2018/09/17 失效的意思啦！报告完毕！无论如何，我们都可以透过 nslookup, host, dig 等等的指令来查询主机名与 IP 的对应，这些指令的用法可以请你以 man command 来查询更多的用法喔！

19.3 DNS 服务器的软件、种类与 cache only DNS 服务器设定

谈完了一些基础概念后，接下来让我们来聊一聊，那如何设定好 DNS 服务器啊？这当然就得由软件安装谈起啦！ 在这个小节，我们先不要谈 DNS 记录的正反解咚咚，只讲到 hint 这个 . (root) 的 zone，谈一谈最简单的仅有快取的 DNS 服务器 (Caching only DNS server) 吧！

19.3.1 架设 DNS 所需要的软件

终于废话都说完了！相信你大概也有点累的吧？鸟哥是蛮累的啦，因为手臂、肩颈酸痛的毛病颇严重....咦！讲这个干嘛？ @_@ 好啦，我们终于要来安装 DNS 所需要的软件了！还记得前面提过的，我们要使用的 DNS 软件就是使用柏克莱大学发展出来的 BIND (Berkeley Internet Name Domain, BIND) 这个啦！那么怎么知道你安装了没？不就是 rpm 与 yum 吗？自己查查看。

```
[root@www ~]# rpm -qa &#124; grep '^bind'
bind-libs-9.7.0-5.P2.el6_0.1.x86_64   &lt;==给 bind 与相关指令使用的函式库
bind-utils-9.7.0-5.P2.el6_0.1.x86_64  &lt;==这个是客户端搜寻主机名的相关指令
bind-9.7.0-5.P2.el6_0.1.x86_64        &lt;==就是 bind 主程序所需软件
bind-chroot-9.7.0-5.P2.el6_0.1.x86_64 &lt;==将 bind 主程序关在家里面！
```

上面比较重要的是那个『 bind-chroot 』啦！所谓的 chroot 代表的是『 change to root(根目录) 』的意思，root 代表的是根目录。早期的 bind 默认将程序启动在 /var/named 当中，但是该程序可以在根目录下的其他目录到处转移，因此若 bind 的程序有问题时，则该程序会造成整个系统的危害。为避免这个问题， 所以我们将某个目录指定为 bind 程序的根目录，由于已经是根目录，所以 bind 便不能离开该目录！所以若该程序被攻击，了不起也是在某个特定目录底下搞破坏而已。 CentOS 6.x 默认将 bind 锁在 /var/named/chroot 目录中喔！

我们主程序是由 bind, bind-chroot 所提供，那前一小节提到的，每部 DNS 服务器都要有的 . (root) 这个 zone file 在哪里？它也是由 bind 所提供的喔！ (CentOS 4.x, 5.x 所提供的 caching-nameserver 软件并不存在 CentOS 6.x 当中了喔！已经被涵盖于 bind 软件内！)

19.3.2 BIND 的默认路径设定与 chroot

要架设好 BIND 需要什么设定数据呢？基本上有两个主要的数据要处理：

- BIND 本身的配置文件：主要规范主机的设定、zone file 的所在、权限的设定等；
- 正反解数据库档案 (zone file)：记录主机名与 IP 对应的等。

BIND 的配置文件为 /etc/named.conf，在这个档案里面可以规范 zone file 的完整檔名喔！ 也就是说，你的 zone file 其实是由 /etc/named.conf 所指定的，所以 zone file 档名可以随便取啦！ 只要 /etc/named.conf 内规范为正确即可。一般来说， CentOS 6.x 的默认目录是这样的：

- /etc/named.conf ：这就是我们的主配置文件啦！
- /etc/sysconfig/named ：是否启动 chroot 及额外的参数，就由这个档案控制；
- /var/named/ ：数据库档案默认放置在这个目录
- /var/run/named ：named 这支程序执行时默认放置 pid-file 在此目录内。

- /etc/sysconfig/named 与 chroot 环境

不过，为了系统的安全性考虑，一般来说目前各主要 distributions 都已经自动的将你的 bind 相关程序给他 chroot 了！ 那你如何知道你 chroot 所指定的目录在哪里呢？其实是记录在 /etc/sysconfig/named 里面啦！你可以先查阅一下：

```
[root@www ~]# cat /etc/sysconfig/named
ROOTDIR=/var/named/chroot
```

事实上该档案内较有意义的就只有上面这一行，意思是说：『我要将 named 给他 chroot ，并且变更的根目录为 /var/named/chroot 』喔！由于根目录已经被变更到 /var/named/chroot 了，但 bind 的相关程序是需要 /etc, /var/named, /var/run ...等目录的，所以实际上咱们 bind 的相关程序所需要的所有数据会是在：

- **/var/named/chroot**/etc/named.conf
- **/var/named/chroot**/var/named/zone_file1
- **/var/named/chroot**/var/named/zone_file.....
- **/var/named/chroot**/var/run/named/...

哇！真是好麻烦～不过，不要太担心！因为新版本的 CentOS 6.x 已经将 chroot 所需要使用到的目录，透过 mount --bind 的功能进行目录链接了 (参考 /etc/init.d/named 内容)，举例来说，我们需要的 /var/named 在启动脚本中透过 mount --bind /var/named /var/named/chroot/var/named 进行目录绑定啰！所以在 CentOS 6.x 当中，你根本无须切换至 /var/named/chroot/ 了！使用正规的目录即可喔！就是这样简单！^_^

Tips: 事实上， /etc/sysconfig/named 是由 /etc/init.d/named 启动时所读入的，所以你也可以直接修改 /etc/init.d/named 这个 script 哩！



19.3.3 单纯的 cache-only DNS 服务器与 forwarding 功能

在下一小节开始介绍正、反解 zone 的数据设定之前，在这个小节当中，我们先来谈一个单纯修改配置文件，而不必设计 zone file 的环境，那就是不具有自己正反解 zone 的仅进行快取的 DNS 服务器。

- 什么是 cache-only 与 forwarding DNS 服务器呢？

有个只需要 . 这个 zone file 的简单 DNS 服务器，我们称这种没有自己公开的 DNS 数据库的服务器为 cache-only (仅快取) DNS server！顾名思义，这个 DNS server 只有快取搜寻结果的功能，也就是说，他本身并没有主机名与 IP 正反解的配置文件，完全是由对外的查询来提供他的数据源！

那如果连 . 都不想要呢？那就得要指定一个上层 DNS 服务器作为你的 forwarding (转递) 目标，将原本自己要往 . 查询的任务，丢给上层 DNS 服务器去烦恼即可。 如此一来，我们这部具有 forwarding 功能的 DNS 服务器，甚至连 . 都不需要了！因为 . 有记录在上层 DNS 上头了嘛！

如同刚刚提到的，cache only 的 DNS 并不存在数据库 (其实还是存在 . 这个 root 领域的 zone file)， 因此不论是谁来查询数据，这部 DNS 一律开始从自己的快取以及 . 找起，整个流程与图 19.1-4 相同。那如果具有 forwarding 功能呢？果真如此，那即使你的 DNS 具有 . 这个 zone file，这部 DNS 还是会将查询权『委请』上层 DNS 查询的，这部 DNS 服务器当场变成客户端啦！查询流程会变这样喔：

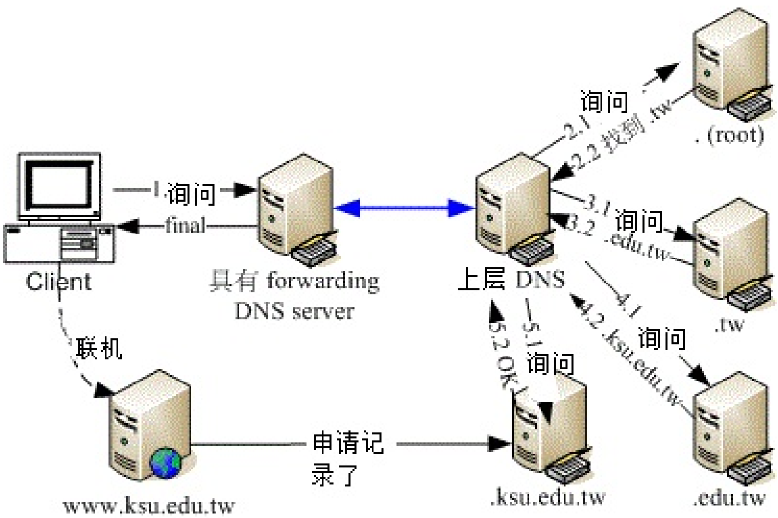


图 19.3-1、具有 forwarding 功能的 DNS 服务器查询方式

观察上图的查询方向，你会发现到，具有 forwarding 机制时，查询权会委请上层 DNS 服务器来处理，所以根本也不需要 . 这个位置所在的 zone 啦。一般来说，如果你的环境需要架设一个 cache-only 的 DNS 服务器时，其实可以直接加上 forwarding 的机制，让查询权指向上层或者是流量较大的上层 DNS 服务器即可。那既然 cache only 的服务器并没有数据库，forwarding 机制甚至不需要 . 的 zone ，那干嘛还得要架设这样的 DNS 呢？是有理由的啦！

- 什么时候有架设 cache-only DNS 的需求？

在某些公司行号里头，为了预防员工利用公司的网络资源作自己的事情，所以都会针对 Internet 的联机作比较严格的限制。当然啦，连 port 53 这个 DNS 会用到的 port 也可能会被挡在防火墙之外的～这个时候，你可以在『防火墙的那部机器上面，加装一个 cache-only 的 DNS 服务！』

这是什么意思呢？很简单啊！就是你自己利用自己的防火墙主机上的 DNS 服务去帮你的 Client 端解译 hostname <--> IP 啰！因为防火墙主机可以设定放行自己的 DNS 功能，而 Client 端就设定该防火墙 IP 为 DNS 服务器的 IP 即可！哈哈！这样就可以取得主机名与 IP 的转译啦！所以，通常架设 cache only DNS 服务器大都是为了系统安全啰。

- 实际设定 cache-only DNS server

那如何在你的 Linux 主机上架设一个 cache-only 的 DNS 服务器呢？其实眞的很简单的啦！因为不需要设定正反解的 zone (只需要 . 的 zone 支持即可)，所以只要设定一个档案 (就是 named.conf 主配置文件) 即可！眞是快乐得不得了呐！ 另外，cache-only 只要加上个 forwarders 的设定即可指定 forwarding 的数据，所以底下我们将设定具有 forwarding 的 cache-only DNS 服务器吧！

---

1. 编辑主要配置文件： /etc/named.conf

   虽然我们具有 chroot 的环境，不过由于 CentOS 6.x 已经透过启动脚本帮我们进行档案与目录的挂载链接，所以请你直接修改 /etc/named.conf 即可呦！不要再去 /var/named/chroot/etc/named.conf 修改啦！ 在这个档案中，主要是定义跟服务器环境有关的设定，以及各个 zone 的领域及数据库所在文件名。 在鸟哥的这个案例当中，因为使用了 forwarding 的机制，所以这个 cache-only DNS 服务器并没有 zone (连 . 都没有)，所以我们只要设定好跟服务器有关的设定即可。设定这个档案的时候请注意：

   - 批注数据是放置在两条斜线『 // 』后面接的数据
   - 每个段落之后都需要以分号『 ; 』来做为结尾！ 鸟哥将这个档案再简化如下的样式：

```
[root@www ~]# cp /etc/named.conf /etc/named.conf.raw
[root@www ~]# vim /etc/named.conf
// 在预设的情况下，这个档案会去读取 /etc/named.rfc1912.zones 这个领域定义档
// 所以请记得要修改成底下的样式啊！
options {
        listen-on port 53  { any; };     //可不设定，代表全部接受
        directory          "/var/named"; //数据库默认放置的目录所在
        dump-file          "/var/named/data/cache_dump.db"; //一些统计信息
        statistics-file    "/var/named/data/named_stats.txt";
        memstatistics-file "/var/named/data/named_mem_stats.txt";
        allow-query        { any; };     //可不设定，代表全部接受
        recursion yes;                   //将自己视为客户端的一种查询模式
        forward only;                    //可暂时不设定
        forwarders {                     //是重点！
                168.95.1.1;              //先用中华电信的 DNS 当上层
                139.175.10.20;           //再用 seednet 当上层
        };
};  //最终记得要结尾符号！
```

鸟哥将大部分的数据都予以删除，只将少部分保留的数据加以小部分的修订而已。在 named.conf 的结构中，与服务器环境有关的是由 options 这个项目内容设定的，因为 options 里面还有很多子参数， 所以就以大括号 { } 包起来啰。至于 options 内的子参数在上面提到的较重要的项目简单叙述如下：

- listen-on port 53 { any; }; 监听在这部主机系统上面的哪个网络接口。预设是监听在 localhost，亦即只有本机可以对 DNS 服务进行查询，那当然是很不合理啊！ 所以这里要将大括号内的数据改写成 any。记得，因为可以监听多个接口，因此 any 后面得要加上分号才算结束喔！ 另外，这个项目如果忘记写也没有关系，因为默认是对整个主机系统的所有接口进行监听的。

- directory "/var/named"; 意思是说，如果此档案底下有规范到正、反解的 zone file 档名时，该档名预设应该放置在哪个目录底下的意思。预设放置到 /var/named/ 底下。由于 chroot 的关系，最终这些数据库档案会被主动链接到 /var/named/chroot/var/named/ 这个目录。

- dump-file, statistics-file, memstatistics-file 与 named 这个服务有关的许多统计信息，如果想要输出成为档案的话，预设的档名就如上所述。鸟哥自己很少看这些统计资料， 所以，这三个设定值写不写应该都是没有关系的。

- allow-query { any; }; 这个是针对客户端的设定，到底谁可以对我的 DNS 服务提出查询请求的意思。原本的档案内容预设是针对 localhost 开放而已， 我们这里改成对所有的用户开放 (当然啦，防火墙也得放行才行)。不过，预设 DNS 就是对所有用户放行，所以这个设定值也可以不用写。

- forward only ; 这个设定可以让你的 DNS 服务器仅进行 forward，即使有 . 这个 zone file 的设定，也不会使用 . 的资料， 只会将查询权交给上层 DNS 服务器而已，是 cache only DNS 最常见的设定了！

- forwarders { 168.95.1.1; 139.175.10.20; } ; 既然有 forward only，那么到底要对哪部上层 DNS 服务器进行转递呢？那就是 forwarders (不要忘记那个 s) 设定值的重要性了！由于担心上层 DNS 服务器也可能会挂点，因此可以设定多部上层 DNS 服务器喔！每一个 forwarder 服务器的 IP 都需要有『 ; 』来做为结尾！ 很简单吧！至于更多的参数我们会在后续篇幅当中慢慢介绍的。这样就已经设定完成了最简单的 cache only DNS server 了！

---

2. 启动 named 并观察服务的埠口

启动总不会忘记吧？赶快去启动一下吧！同时启动完毕之后，观察一下由 named 所开启的埠口，看看到底哪些埠口会被 DNS 用到的！

```
# 1\. 启动一下 DNS 这玩意儿！
[root@www ~]# /etc/init.d/named start
Starting named:                           [  OK  ]
[root@www ~]# chkconfig named on

# 2\. 到底用了多少埠口呢？
[root@www ~]# netstat -utlnp &#124; grep named
Proto Recv-Q Send-Q Local Address        Foreign Address  State  PID/Program name
tcp        0      0 192.168.100.254:53  0.0.0.0:*        LISTEN 3140/named
tcp        0      0 192.168.1.100:53    0.0.0.0:*        LISTEN 3140/named
tcp        0      0 127.0.0.1:53        0.0.0.0:*        LISTEN 3140/named
tcp        0      0 127.0.0.1:953       0.0.0.0:*        LISTEN 3140/named
tcp        0      0 ::1:953             :::*             LISTEN 3140/named
udp        0      0 192.168.100.254:53  0.0.0.0:*               3140/named
udp        0      0 192.168.1.100:53    0.0.0.0:*               3140/named
udp        0      0 127.0.0.1:53        0.0.0.0:*               3140/named
```

我们知道 DNS 会同时启用 UDP/TCP 的 port 53，而且是针对所有接口，因此上面的数据并没有什么特异的部分。不过，怎么会有 port 953 且仅针对本机来监听呢？其实那是 named 的远程控制功能，称为远程名称解析服务控制功能 (remote name daemon control, rndc)。预设的情况下，仅有本机可以针对 rndc 来控制。我们会在后续的章节再来探讨这个 rndc 啦，目前我们只要知道 UDP/TCP port 53 有启动即可。

---

3. 检查 /var/log/messages 的内容讯息 (极重要！)

named 这个服务的记录文件就直接给他放置在 /var/log/messages 里面啦，所以来看看里面的几行登录信息吧！

```
[root@www ~]# tail -n 30 /var/log/messages &#124; grep named
Aug  4 14:57:09 www named[3140]: starting BIND 9.7.0-P2-RedHat-9.7.0-5.P2.el6_0.1 -u
 -t /var/named/chroot &lt;==说明的是 chroot 在哪个目录下！
Aug  4 14:57:09 www named[3140]: adjusted limit on open files from 1024 to 1048576
Aug  4 14:57:09 www named[3140]: found 1 CPU, using 1 worker thread
Aug  4 14:57:09 www named[3140]: using up to 4096 sockets
Aug  4 14:57:09 www named[3140]: loading configuration from '/etc/named.conf'
Aug  4 14:57:09 www named[3140]: using default UDP/IPv4 port range: [1024, 65535]
Aug  4 14:57:09 www named[3140]: using default UDP/IPv6 port range: [1024, 65535]
Aug  4 14:57:09 www named[3140]: listening on IPv4 interface lo, 127.0.0.1#53
Aug  4 14:57:09 www named[3140]: listening on IPv4 interface eth0, 192.168.1.100#53
Aug  4 14:57:09 www named[3140]: listening on IPv4 interface eth1, 192.168.100.254#53
Aug  4 14:57:09 www named[3140]: generating session key for dynamic DNS
Aug  4 14:57:09 www named[3140]: command channel listening on 127.0.0.1#953
Aug  4 14:57:09 www named[3140]: command channel listening on ::1#953
Aug  4 14:57:09 www named[3140]: the working directory is not writable
Aug  4 14:57:09 www named[3140]: running
```

上面最重要的是第一行出现的『-t ...』那个项目指出你的 chroot 目录啰。另外，上面表格中特殊字体的部分，有写到读取 /etc/named.conf，代表可以顺利的加载 /var/named/etc/named.conf 的意思。如果上面有出现冒号后面接数字 (:10)， 那就代表某个档案内的第十行有问题的意思，届时再进入处理即可。要注意的是，即使 port 53 有启动，但有可能 DNS 服务是错误的，此时这个登录档就显的非常重要！每次重新启动 DNS 后，请务必查阅一下这个档案的内容！！

Tips: 如果你在 /var/log/messages 里面一直看到这样的错误信息： couldn't add command channel 127.0.0.1#953: not found 那表示你还必需要加入 rndc key ，请参考本章后面的 利用 RNDC 指令管理 DNS 服务器 的介绍，将他加入你的 named.conf 中！



1. 测试：

   如果你的 DNS 伺服器具有连上因特网的功能，那么透过『 dig www.google.com @127.0.0.1 』这个基本指令执行看看， 如果有找到 google 的 IP ，并且输出数据的最底下显示『 SERVER: 127.0.0.1#53(127.0.0.1) 』的字样， 那就代表应该是成功啦！其他更详细的测试请参考：19.2 小节的内容

2. 特别说明：Forwarders 的好处与问题分析

关于 forwarder 的好处与坏处，其实有很多种的意见！大致的意见可分为这两派：

- 利用 Forwarder 的功能来增进效能的理论：

  这些朋友们认为，当很多的下层 DNS 服务器都使用 forwarder 时，那么那个被设定为 forwarder 的主机，由于会记录很多的查询信息记录 (请参考图 19.1-4 的说明)，因此，对于那些下层的 DNS 服务器而言，查询速度会增快很多，亦即会节省很多的查询时间！因为 forwarder 服务器里面有较多的快取记录了， 所以包括 forwarder 本身，以及所有向这部 forwarder 要求数据的 DNS 服务器，都能够减少往 . 查询的机会， 因此速度当然增加。

- 利用 Forwarder 反而会使整体的效能降低：

  但是另外一派则持相反的见解！这是因为当主 DNS 本身的『业务量』就很繁忙的时候，那么你的 cache only DNS 服务器还向他要求数据，因为他原本的数据传输量就太大了，带宽方面可能负荷不量，而太多的下层 DNS 还向他要求数据，所以他的查询速度会变慢！因为查询速度变慢了，而你的 cache only server 又是向他提出要求的，所以自然两边的查询速度就会同步下降！

很多种说法啦！鸟哥本人也觉得很有趣哩！只是不知道哪一派较正确就是了，不过可以知道的是，如果上层的 DNS 速度很快的话，那么他被设定为 forwarder 时，或许真的可以增加不少效能哩！

19.4 DNS 服务器的详细设定

好了，经过上面的说明后，我们大概知道 DNS 的几个小细节是这样的：

1. DNS 服务器的架设需要上层 DNS 的授权才可以成为合法的 DNS 服务器 (否则只是练功)；
2. 配置文件位置：目前 bind 程序已进行 chroot，相关目录可参考 /etc/sysconfig/named；
3. named 主要配置文件是 /etc/named.conf；
4. 每个正、反解领域都需要一个数据库档案，而文件名则是由 /etc/named.conf 所设定；
5. 当 DNS 查询时，若本身没有数据库档案，则前往 root (.) 或 forwarders 服务器查询；
6. named 是否启动成功务必要查阅 /var/log/messages 内的信息！

其中第一点很重要，因为我们尚未向上层 ISP 注册合法的领域名，所以我们当然就没有权利架设合法的 DNS 服务器了。 而由于担心我们的 DNS 服务器会与外部因特网环境互相干扰，所以底下鸟哥将主要以一个 centos.vbird 的领域名来架设 DNS 服务器，如此一来咱们就可以好好的玩一玩自己局域网络内的 DNS 啦！

19.4.1 正解文件记录的数据 (Resource Record, RR)

既然 DNS 最早之前的目的就是要从主机名去找到 IP，所以就让我们先从正解 zone 来谈起吧。既然要谈正解， 那么就应该要了解正解档案记录的信息有哪些吧？在这个小节里面，我们就先来谈谈正解 zone 常常记录的数据有哪些吧。

- 正解文件资源记录 (resource record, RR) 格式

我们从前面几个小节的 dig 指令输出结果中，可以发现到一个有趣的咚咚，那就是输出的数据格式似乎是固定的！ 举例来说，查询 www.ksu.edu.tw 的 IP 时，输出的结果为：

```
[root@www ~]# dig www.ksu.edu.tw
....(前面省略)....
;; ANSWER SECTION:
www.ksu.edu.tw.         2203    IN      A       120.114.100.101

;; AUTHORITY SECTION:
ksu.edu.tw.             911     IN      NS      dns1.ksu.edu.tw.
....(后面省略)....
# 上面的输出数据已经被简化过了，重点是要大家了解 RR 的格式
```

在答案的输出阶段，主要查询得到的是 A 的标志，在认证阶段，则是提供 ksu.edu.tw 的 NS 服务器为哪一部的意思。 格式非常接近，只是 A 后面接 IP，而 NS 后面接主机名而已。我们可以将整个输出的格式简化成为如下的说明：

```
[domain]   [ttl]          IN [[RR type]  [RR data]]
[待查数据] [暂存时间(秒)] IN [[资源类型] [资源内容]]
```

上表中，关键词 IN 是固定的，而 RR type 与 RR data 则是互有关连性的，例如刚刚才提过的 A 就是接 IP 而不是主机名啊。此外，在 domain 的部分，若可能的话，请尽量使用 FQDN，亦即是主机名结尾加上一个小数点的 (.) 就被称为 FQDN 了！例如刚刚 dig www.ksu.edu.tw 的输出结果中，在答案阶段时，搜寻的主机名会变成 www.ksu.edu.tw. 喔！注意看最后面有个小数点喔！那个小数点非常重要！

至于 ttl 就是 time to live 的缩写，意思就是当这笔记录被其他 DNS 服务器查询到后， 这个记录会保持在对方 DNS 服务器的快取中，保持多少秒钟的意思。所以，当你反复执行 dig www.ksu.edu.tw 之后，就会发现这个时间会减少！为什么呢？因为在你的 DNS 快取中，这笔数据能够保存的时间会开始倒数， 当这个数字归零后，下次有人再重新搜寻这笔记录时，你的 DNS 就会重新沿着 . (root) 开始重来搜寻一遍， 而不会从快取里面捉取了 (因为快取内的资料会被舍弃)。

由于 ttl 可由特定的参数来统一控管，因此在 RR 的记录格式中，通常这个 ttl 的字段是可以忽略的。 那么常见的 RR 有哪些呢？我们将正解文件的 RR 记录格式汇整如下：

```
# 常见的正解文件 RR 相关信息
[domain]    IN  [[RR type]  [RR data]]
主机名.   IN  A           IPv4 的 IP 地址
主机名.   IN  AAAA        IPv6 的 IP 地址
领域名.   IN  NS          管理这个领域名的服务器主机名字.
领域名.   IN  SOA         管理这个领域名的七个重要参数(容后说明)
领域名.   IN  MX          顺序数字  接收邮件的服务器主机名字
主机别名.   IN  CNAME       实际代表这个主机别名的主机名字.
```

接下来我们以昆山科大的 DNS 设定，包括 ksu.edu.tw 这个领域 (domain, zone)，以及 www.ksu.edu.tw 这个主机名 (FQDN) 的查询结果来跟大家解释每个 RR 记录的信息为何吻！

- A, AAAA ：查询 IP 的记录

这个 A 的 RR 类型是在查询某个主机名的 IP，也是最长被查询的一个 RR 标志喔！举例来说，要找到 www.ksu.edu.tw 的 A 的话，就是这样查：

```
[root@www ~]# dig [-t a] www.ksu.edu.tw
;; ANSWER SECTION:
www.ksu.edu.tw.         2987    IN      A       120.114.100.101
# 主机FQDN.             ttl                     这部主机的 IP 就是这里
# 仅列出答案阶段的资料，后续的 RR 相关标志也是这样显示的喔！
# 指令列中的 [-t a] 可以不加，而最左边主机名结尾都会有小数点喔！
```

左边是主机名，当然，你也可以让你的 domain 拥有一个 A 的标志，例如『 dig google.com 』也能找到 IP。 不过，咱们昆山科大的 ksu.edu.tw 则没有设定 IP 就是了。要再次特别强调的，主机名如果是全名， 结尾部分请务必加上小数点。如果你的 IP 设定的是 IPv6 的话，那么查询就得要使用 aaaa 类型才行。

- NS ：查询管理领域名 (zone) 的服务器主机名

如果你想要知道 www.ksu.edu.tw 的这笔记录是由哪部 DNS 服务器提供的，那就得要使用 NS (NameServer) 的 RR 类型标志来查询。不过，由于 NS 是管理整个领域的，因此，你得要查询的目标将得输入 domain，亦即 ksu.edu.tw 才行喔！举例如下：

```
[root@www ~]# dig -t ns ksu.edu.tw
;; ANSWER SECTION:
ksu.edu.tw.            1596    IN      NS  dns1.ksu.edu.tw.

;; ADDITIONAL SECTION:
dns1.ksu.edu.tw.       577     IN      A   120.114.50.1
# 除了列出 NS 是哪部服务器之外，该服务器的 IP 也会额外提供！
```

前面提过，DNS 服务器是很重要的，因此至少都会有两部以上。昆山科大共有三部 DNS 服务器，鸟哥仅列出第一部提供参考。 NS 后面会加服务器名称，而这个服务器的 IP 也会额外提供才对！因此 NS 经常伴随 A 的标志啊！这样你才能到 NS 去查询数据嘛！这样说有理解吧？ ^_^

- SOA ：查询管理领域名的服务器管理信息

如果你有多部 DNS 服务器管理同一个领域名时，那么最好使用 master/slave 的方式来进行管理。既然要这样管理， 那就得要宣告被管理的 zone file 是如何进行传输的，此时就得要 SOA (Start Of Authority) 的标志了。先来瞧瞧昆山科大的设定是怎样：

```
[root@www ~]# dig -t soa ksu.edu.tw
;; ANSWER SECTION:
ksu.edu.tw.       3600   IN     SOA    dns1.ksu.edu.tw.   abuse.mail.ksu.edu.tw.
  2010080369 1800 900 604800 86400
# 上述的输出结果是同一行喔！
```

SOA 主要是与领域有关，所以前面当然要写 ksu.edu.tw 这个领域名。而 SOA 后面共会接七个参数，这七个参数的意义依序是：

1. Master DNS 服务器主机名：这个领域主要是哪部 DNS 作为 master 的意思。在本例中， dns1.ksu.edu.tw 为 ksu.edu.tw 这个领域的主要 DNS 服务器啰；
2. 管理员的 email：那么管理员的 email 为何？发生问题可以联络这个管理员。要注意的是， 由于 @ 在数据库档案中是有特别意义的，因此这里就将 abuse@mail.ksu.edu.tw 改写成 abuse.mail.ksu.edu.tw ，这样看的懂了吗？
3. 序号 (Serial)：这个序号代表的是这个数据库档案的新旧，序号越大代表越新。 当 slave 要判断是否主动下载新的数据库时，就以序号是否比 slave 上的还要新来判断，若是则下载，若不是则不下载。 所以当你修订了数据库内容时，记得要将这个数值放大才行！ 为了方便用户记忆，通常序号都会使用日期格式『YYYYMMDDNU』来记忆，例如昆山科大的 2010080369 序号代表 2010/08/03 当天的第 69 次更新的感觉。不过，序号不可大于 2 的 32 次方，亦即必须小于 4294967296 才行喔。

4. 更新频率 (Refresh)：那么啥时 slave 会去向 master 要求数据更新的判断？ 就是这个数值定义的。昆山科大的 DNS 设定每 1800 秒进行一次 slave 向 master 要求数据更新。那每次 slave 去更新时， 如果发现序号没有比较大，那就不会下载数据库档案。

5. 失败重新尝试时间 (Retry)：如果因为某些因素，导致 slave 无法对 master 达成联机，那么在多久的时间内，slave 会尝试重新联机到 master。在昆山科大的设定中，900 秒会重新尝试一次。意思是说，每 1800 秒 slave 会主动向 master 联机，但如果该次联机没有成功，那接下来尝试联机的时间会变成 900 秒。若后来有成功，则又会恢复到 1800 秒才再一次联机。

6. 失效时间 (Expire)：如果一直失败尝试时间，持续联机到达这个设定值时限， 那么 slave 将不再继续尝试联机，并且尝试删除这份下载的 zone file 信息。昆山科大设定为 604800 秒。意思是说，当联机一直失败，每 900 秒尝试到达 604800 秒后，昆山科大的 slave 将不再更新，只能等待系统管理员的处理。

7. 快取时间 (Minumum TTL)：如果这个数据库 zone file 中，每笔 RR 记录都没有写到 TTL 快取时间的话，那么就以这个 SOA 的设定值为主。

除了 Serial 不可以超过 2 的 32 次方之外，有没有其它的限制啊针对这几个数值？是有的，基本上就是这样：

- Refresh >= Retry *2
- Refresh + Retry < Expire
- Expire >= Rrtry * 10
- Expire >= 7Days

一般来说，如果 DNS RR 资料变更情况频繁的，那么上述的相关数值可以订定的小一些，如果 DNS RR 是很稳定的， 为了节省带宽，则可以将 Refresh 设定的较大一些。

- CNAME ：设定某主机名的别名 (alias)

有时候你不想要针对某个主机名设定 A 的标志，而是想透过另外一部主机名的 A 来规范这个新主机名时， 可以使用别名 (CNAME) 的设定喔！举例来说，追踪 www.google.com 时，你会发现这样：

```
[root@www ~]# dig www.google.com
;; ANSWER SECTION:
www.google.com.         557697  IN      CNAME   www.l.google.com.
www.l.google.com.       298     IN      A       72.14.203.99
```

意思是说，当你要追查 www.google.com 时，请找 www.1.google.com 那个主机，而那个主机的 A 就上面第二行的显示了。 鸟哥常常开玩笑的说，你知道鸟哥的身份证字号吗？你到户政事务所去查『鸟哥』时，他会说：『没这个人啊！因为没有人姓鸟...』， 这个『鸟哥』就是别名 (CNAME) ，而对应到的名称就是『蔡某某』，这个蔡某某才真的有身份字号的意思～一层一层去追踪啰～

这个 CNAME 有啥好处呢？用 A 就好了吧？其实还是有好处的，举例来说，如果你有一个 IP，这个 IP 是给很多主机名使用的。 那么当你的 IP 更改时，所有的数据就得通通更新 A 标志才行。如果你只有一个主要主机名设定 A，而其他的标志使用 CNAME 时，那么当 IP 更改，那你只要修订一个 A 的标志，其他的 CNAME 就跟着变动了！处理起来比较容易啊！

- MX ：查询某领域名的邮件服务器主机名

MX 是 Mail eXchanger (邮件交换) 的意思，通常你的整个领域会设定一个 MX ，代表，所有寄给这个领域的 email 应该要送到后头的 email server 主机名上头才是。先看看昆大的资料：

```
[root@www ~]# dig -t mx ksu.edu.tw
;; ANSWER SECTION:
ksu.edu.tw.             3600    IN      MX      8 mx01.ksu.edu.tw.

;; ADDITIONAL SECTION:
mx01.ksu.edu.tw.        3600    IN      A       120.114.100.28
```

上头的意思是说，当有信件要送给 ksu.edu.tw 这个领域时，则预先将信件传送给 mx01.ksu.edu.tw 这部邮件服务器管理， 当然啦，这部 mx01.ksu.edu.tw 自然就是昆大自己管理的邮件服务器才行！MX 后面接的主机名通常就是合法 mail server， 而想要当 MX 服务器，就得要有 A 的标志才行～所以上表后面就会出现 mx01.ksu.edu.tw 的 A 啊！

那么在 mx01 之前的 8 是什么意思？由于担心邮件会遗失，因此较大型的企业会有多部这样的上层邮件服务器来预先收受信件。 那么到底哪部邮件主机会先收下呢？就以数字较小的那部优先啰！举例来说，如果你去查 google.com 的 MX 标志， 就会发现他有 5 部这样的服务器呢！

19.4.2 反解文件记录的 RR 数据

讲完了正解来谈谈反解吧！在讲反解之前，先来谈谈正解主机名的追踪方式。以 www.ksu.edu.tw. 来说，整个网域的概念来看， 越右边出现的名称代表网域越大！举例来说，.(root) > tw > edu 以此类推。因此追踪时，是由大范围找到小范围， 最后，我们就知道追踪的方向如图 19.1-4 所示那样。

但是 IP 则不一样啊！以昆大的 120.114.100.101 来说好了，当然是 120 > 114 > 100 > 101 ，左边的网域最大！ 与预设的 DNS 从右边向左边查询不一样啊！那怎办？为了解决这个问题，所以反解的 zone 就必须要将 IP 反过来写，而在结尾时加上 .in-addr.arpa. 的结尾字样即可。所以，当你想要追踪反解时，那么反解的结果就会是：

```
[root@www ~]# dig -x 120.114.100.101
;; ANSWER SECTION:
101.100.114.120.in-addr.arpa. 3600 IN   PTR     www.ksu.edu.tw.
```

例如上述的结果中，我们要查询的主机名竟然变成了 IP 反转的模样！所以才称为反解嘛！而反解的标志最重要的就是 PTR 了！

- PTR ：就是反解啊！所以是查询 IP 所对应的主机名

进行反解时，要注意的就是 zone 的名称了！要将 IP 反转过来写，并且结尾加上 .in-addr.arpa. 才行！ 例如 120.114.100.0/24 这个 class C IP 网段的反解设定，就必须要写成：100.114.120.in-addr.arpa. 这样的 zone 名称才行。 而 PTR 后面接的自然就是主机名啰！

在反解最重要的地方就是：后面的主机名尽量使用完整 FQDN，亦即加上小数点 (.) ！为什么呢？举 100.114.120.in-addr.arpa. 为例，如果你只是填写主机名，并没有填写领域名， 那么当人家追踪你的主机名时，你的主机名会变成： www.100.114.120.in-addr.arpa. 的怪模样。这是比较需要注意的地方。

Tips: 老实说，鸟园讨论区的一些有经验的朋友一直在讲，如果担心会有误解，主机名的设定则通通记得是要填写 FQDN 就是了！ 这样绝对不会有问题！ ^_^



19.4.3 步骤一：DNS 的环境规划：正解、反解 zone 的预先定义案例说明

现在假设鸟哥的区网环境中想要设定 DNS 服务器，鸟哥的区网原本规划的域名就是 centos.vbird，且搭配的 IP 网段为 192.168.100.0/24 这一段，因此主要的正解网域为 centos.vbird，而反解的网域则为 192.168.100.0/24， 鸟哥的这部 DNS 服务器想要自己找寻 .(root) 而不透过 forwarders 的辅助，因此还得要 . 的领域正解档。 综合起来说，鸟哥需要设定到的档案就有这几个：

1. named.conf (主要配置文件)
2. named.centos.vbird (主要的 centos.vbird 的正解檔)
3. named.192.168.100 (主要的 192.168.100.0/24 的反解檔)
4. named.ca (由 bind 软件提供的 . 正解檔)

如果我还想要加入其他的领域，例如 niki.vbird 可不可以啊？当然可以啊！就再多一个数据库正解档案即可！ 还有，鸟哥上头这个设定资料为内部私有的，所以你可以完全照着玩！并不会影响到外部的因特网啦！ 只是，因特网也查不到你的 DNS 设定就是了～反正是练功嘛！ ^_^

至于数据库的正、反解对应上，依据实际的测试环境，规划如下 (亦请参考第三章图 3.2-1)：

操作系统与IP	主机名与 RR 标志	说明
Linux (192.168.100.254)	master.centos.vbird (NS, A) www.centos.vbird (A) linux.centos.vbird (CNAME) ftp.centos.vbird (CNAME) forum.centos.vbird (CNAME) www.centos.vbird (MX)	DNS 设置是使用 master.centos.vbird 这个 DNS 服务器名称。至于这部主机的另一个主要名称是 www.centos.vbird，其他的都是 CNAME，这样未来比较好修改。同时给予一个 MX 的标志给主要主机名喔
Linux (192.168.100.10)	slave.centos.vbird (NS, A) clientlinux.centos.vbird(A)	未来作为 slave DNS 的接班人～
WinXP (192.168.1.101)	workstation.centos.vbird (A)	一部经常用来工作的工作机
WinXP (192.168.100.20)	winxp.centos.vbird (A)	一部用来测试的 Windows XP
Win7 (192.168.100.30)	win7.centos.vbird (A)	一部用来测试的 Windows 7

请特别留意啊，一个 IP 可以对应给多个主机名，同样的，一个主机名可以给予多个 IP 喔！主要是因为那部 www.centos.vbird 的机器未来的用途相当的多，鸟哥希望那一部主机有多个名称，以方便未来额外的规划啊。 所以就对该 IP 对应了四个主机名啊！

Tips: 在自家设的没有经过合法授权的 DNS 最好不要以 Internet 上面已经存在的领域名来练习架设！ 举例来说，假设今天你以 192.168.100.254 那部机器来架设 *.yahoo.com 的领域，因为我将 192.168.100.254 放置在第一位，导致每次的查询其实 yahoo.com 这个领域的数据都是直接由 192.168.100.254 所提供，这很不好～因为可能会造成你的客户端的不便～



19.4.4 步骤二：主配置文件 /etc/named.conf 的设置

这个配置文件较多的 options 参数我们已经在 19.3.3 里面谈过，在我们目前的案例中， 则必须要将 forwarders 相关功能取消，并加上禁止传输 zone file 的参数即可。至于 zone 的设定上，必须要包含上个小节谈到的三个主要的 zone 喔！因此这个档案的任务是：

- options：规范 DNS 服务器的权限 (可否查询、forward 与否等)；
- zone：设定出 zone (domain name) 以及 zone file 的所在 (包含 master/slave/hint)；
- 其他：设定 DNS 本机管理接口以及其相关的密钥档案 (key file)。(本章稍后进阶应用再谈)

那就直接看一下鸟哥的范本吧：

```
[root@www ~]# vim /etc/named.conf
options {
        directory       "/var/named";
        dump-file       "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
        memstatistics-file "/var/named/data/named_mem_stats.txt";
        allow-query     { any; };
        recursion yes;
 allow-transfer  { none; };   // 不许别人进行 zone 转移
};

zone "." IN {
        type hint;
        file "named.ca";
};
zone "centos.vbird" IN {             // 这个 zone 的名称
        type master;                 // 是什么类型
        file "named.centos.vbird";   // 档案放在哪里
};
zone "100.168.192.in-addr.arpa" IN {
        type master;
        file "named.192.168.100";
};
```

在 options 里面仅新增一个新的参数，就是 allow-transfer，意义为：

- allow-transfer { none; }; 是否允许来自 slave DNS 对我的整个领域数据进行传送？这个设定值与 master/slave DNS 服务器之间的数据库传送有关。除非你有 slave DNS 服务器，否则这里不要开放喔！因此这里我们先设定为 none。

至于在 zone 里面的设定值，主要则有底下几个：

zone 内的相关参数说明	
设定值	意义
type	该 zone 的类型，主要的类型有针对 . 的 hint，以及自己手动修改数据库档案的 master，与可自动更新数据库的 slave。
file	就是 zone file 的檔名啊！(注意 chroot 与否呦！)
反解 zone	主要就是 in-addr.arpa 这个玩意儿！请参考 19.4.2 的解释

为何档名都是 named 开头呢？这只是个习惯而已，你也可以依据自己的习惯来订定档名的。经过上面的说明，所以我们会知道，zone file 档名都是透过 named.conf 这个配置文件来规范的啊！

19.4.5 步骤三：最上层 . (root) 数据库档案的设定

从 图 19.1-4 可以知道 . 的重要性！那么这个 . 在哪里呢？事实上，它是由 INTERNIC 所管理维护的，全世界共有 13 部管理 . 的 DNS 服务器呢！相关的最新设定在：

- ftp://rs.internic.net/domain/named.root

要不要下载最新的资料随你便，因为我们的 CentOS 6.x 内的 bind 软件已经提供了一个名为 named.ca 的档案了，鸟哥是直接使用系统提供的数据啦。这个档案的内容有点像这样：

```
[root@www ~]# vim /var/named/named.ca
. &lt;==这里有个小数点      518400  IN      NS      A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.      3600000 IN      A       198.41.0.4
# 上面这两行是成对的！代表点由 A.ROOT-SERVERS.NET. 管理，并附上 IP 查询
. &lt;==这里有个小数点      518400  IN      NS      M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.      3600000 IN      A       202.12.27.33
M.ROOT-SERVERS.NET.      3600000 IN      AAAA    2001:dc3::35
# 上面这三行是成对的，代表 M 开头的服务器有 A 与 AAAA 的记录
```

相关的正解标志 NS, A, AAAA 意义，请回 19.4.1 去查询，这里不再解释。 比较特殊的是，由于考虑 IPv6 未来的流行性，因此很多部 . 服务器都加上 AAAA 的 IPv6 功能啰。 这个档案的内容你不要修改啊～因为这个内容是 Internet 上面通用的数据，一般来说，也不会常常变动， 所以不需要更动他，将他放置到正确的目录并改成你所指定的档名即可啊！接下来可以看看其他正解档案啦！

19.4.6 步骤四：正解数据库档案的设定

再来开始正解档的设定吧！正解文件一定要有的 RR 标志有底下几个喔：

- 关于本领域的基础设定方面：例如快取记忆时间 (TTL)、领域名 (ORIGIN) 等；
- 关于 master/slave 的认证方面 (SOA)；
- 关于本领域的领域名服务器所在主机名与 IP 对应 (NS, A)；
- 其他正反解相关的资源记录 (A, MX, CNAME 等)。

相关的 RR 意义请回 19.4.1 去查询。此外，这个档案的特殊符号也得跟大家报告一下：

字符	意义
一定从行首开始	所有设定数据一定要从行首开始，前面不可有空格符。若有空格符，代表延续前一个 domain 的意思～非常重要～
@	这个符号代表 zone 的意思！例如写在 named.centos.vbird 中，@ 代表 centos.vbird.，如果写在 named.192.168.100 档案中，则 @ 代表 100.168.192.in-addr.arpa. 的意思 (参考 named.conf 内的 zone 设定)
.	这个点 (.) 很重要！因为他代表一个完整主机名 (FQDN) 而不是仅有 hostname 而已。举例来说，在 named.centos.vbird 当中写 www.centos.vbird 则代表 FQDN 为 www.centos.vbird.@ ==> www.centos.vbird.centos.vbird. 喔！因此当然要写成 www.centos.vbird. 才对！
;	代表批注符号～似乎 # 也是批注～两个符号都能使用

鸟哥打算沿用系统提供的一些配置文件，然后据以修改成为鸟哥自己需要的环境。整个 DNS 是由 master.centos.vbird 这部服务器管理的，而管理者的 email 为 vbird@www.centos.vbird 这个。整个正解档最终有点像这样：

```
[root@www ~]# vim /var/named/named.centos.vbird
# 与整个领域相关性较高的设定包括 NS, A, MX, SOA 等标志的设定处！
$TTL    600
@                       IN &lt;u&gt;SOA   master.centos.vbird. vbird.www.centos.vbird. (&
                                 &lt;u&gt;2011080401 3H 15M 1W 1D )&lt;/u&gt; ; 与上面是同-
@                       IN &lt;u&gt;NS    master.centos.vbird.&lt;/u&gt;  ; DNS 服务器名称
master.centos.vbird.    IN &lt;u&gt;A     192.168.100.254&lt;/u&gt;;         ; DNS 服务器 I
@                       IN &lt;u&gt;MX 10 www.centos.vbird.&lt;/u&gt;;     ; 领域名的邮件服务

# 针对 192.168.100.254 这部主机的所有相关正解设定。
www.centos.vbird.       IN &lt;u&gt;A     192.168.100.254&lt;/u&gt;;
linux.centos.vbird.     IN CNAME www.centos.vbird.
ftp.centos.vbird.       IN CNAME www.centos.vbird.
forum.centos.vbird.     IN CNAME www.centos.vbird.

# 其他几部主机的主机名正解设定。
slave.centos.vbird.       IN A    192.168.100.10
clientlinux.centos.vbird. IN A    192.168.100.10
workstation.centos.vbird. IN A    192.168.1.101
winxp.centos.vbird.       IN A    192.168.100.20
&lt;u&gt;win7                          IN A    192.168.100.30&lt;/u&gt;;  ; 这是简化的写法！
```

再次强调，一个正解的数据库设定中，至少应该要有 $TTL, SOA, NS (与这部 NS 主机名的 A)， 鸟哥将这些基本要用到的标志写在上表的第一部份。至于其他的，则是相关的主机名正解设定啰。 如果这些设定值你看不懂，那么，可以肯定的是，请回 19.4.1 去瞧瞧吧！ 底下强调一下之前没有讲到的设定值项目：

关于本领域的一些设定值	
设定值	说明
$TTL	为了简化每笔 RR 记录的设定，因此我们将 TTL 挪到最前面统一设定。因为鸟哥的 DNS 服务器还在测试中，所以 TTL 写了个比较小的数值，可以存在对方 DNS 服务器的快取 600 秒而已。
$ORIGIN	这个设定值可以重新指定 zone 的定义。在预设的情况下，这个正反解数据库档案中的 zone 是由 named.conf 所指定的，就是 zone 那个参数的功能。不过，这个 zone 是可以改的，就是用 $ORIGIN 来修订就是了。通常这个设定值不会用到的

老实说，初次设定 DNS 的朋友大概都会被那个小数点 (.) 玩死～其实你不要太紧张，只要记住：『 加上了 . 表示这是个完整的主机名 (FQDN)，亦即是 "hostname + domain name" 了，如果没有加上 . 的话，表示该名称仅为 "hostname" 而已！因为我们这个配置文件的 zone 是 centos.vbird， 所以上表的最后一行，鸟哥只写出主机名 (win7) ，因为没有小数点结尾，因此完整的 FQDN 要加上 zone，所以主机名 win7 代表的是： win7.centos.vbird. 喔！

19.4.7 步骤五：反解数据库档案的设定

反解跟正解一样，还都需要 TTL, SOA, NS 等等的，但是相对于正解里面有 A，反解里面则仅有 PTR 喔！ 另外，由于反解的 zone 名称是很怪 zz.yy.xx.in-addr.arpa. 的模样，因此只要在反解里面要用到主机名时， 务必使用 FQDN 来设定啊！更多与反解有关的资料，请到 19.4.2 去查阅喔！ 至于 192.168.100.0/24 这个网域的 DNS 反解则成为：

```
[root@www ~]# vim /var/named/named.192.168.100
$TTL    600
@       IN SOA  master.centos.vbird. vbird.www.centos.vbird. (
                2011080401 3H 15M 1W 1D )
@       IN NS   master.centos.vbird.
254     IN PTR  master.centos.vbird.  ; 将原本的 A 改成 PTR 的标志而已

254     IN PTR  www.centos.vbird.     ; 这些是特定的 IP 对应
10      IN PTR  slave.centos.vbird.
20      IN PTR  winxp.centos.vbird.
30      IN PTR  win7.centos.vbird.

101     IN PTR  dhcp101.centos.vbird.  ; 可能针对 DHCP (第十二章) 的 IP 设定
102     IN PTR  dhcp102.centos.vbird.
....(中间省略)....
200     IN PTR  dhcp200.centos.vbird.
```

因为我们的 zone 是 100.168.192.in-addr.arpa. 这一个，因此 IP 的全名部分已经含有 192.168.100 了， 所以在上表当中的最左边，数值只需要存在最后一个 IP 即可。因此 254 就代表 192.168.100.254 啰！ 此外，为了担心 DHCP 自动分配的 IP 没有对应的主机名，所以这里也附挂了 192.168.100.{101~200} 的主机名对应喔！

19.4.8 步骤六：DNS 的启动、观察与防火墙

DNS 的启动也太简单了吧？就直接利用系统提供的启动 script 即可！

```
[root@www ~]# /etc/init.d/named restart  &lt;==也可能是需要 restart 喔
[root@www ~]# chkconfig named on
```

但即使画面上出现的是『确定』或『OK』，都不见得你的 DNS 服务是正常的。所以，请你『务必』查阅 /var/log/messages 的内容才行！基本上，内容会有点像这样：

```
[root@www ~]# tail -n 30 /var/log/messages &#124; grep named
named[3511]: starting BIND 9.7.0-P2-RedHat-9.7.0-5.P2.el6_0.1 -u named -t
/var/named/chroot
named[3511]: adjusted limit on open files from 1024 to 1048576
named[3511]: found 1 CPU, using 1 worker thread
named[3511]: using up to 4096 sockets
named[3511]: loading configuration from '/etc/named.conf'
named[3511]: using default UDP/IPv4 port range: [1024, 65535]
named[3511]: using default UDP/IPv6 port range: [1024, 65535]
named[3511]: listening on IPv4 interface lo, 127.0.0.1#53
named[3511]: listening on IPv4 interface eth0, 192.168.1.100#53
named[3511]: listening on IPv4 interface eth1, 192.168.100.254#53
named[3511]: command channel listening on 127.0.0.1#953
named[3511]: command channel listening on ::1#953
named[3511]: the working directory is not writable
named[3511]: zone 100.168.192.in-addr.arpa/IN: loaded serial 2011080401
named[3511]: zone centos.vbird/IN: loaded serial 2011080401
named[3511]: running
```

上面的输出讯息中，你得要特别注意有画底线的部分。包括 -t chroot_dir 是设定 chroot 目录的位置， 而配置文件 (configuration) 则是 /etc/named.conf，最重要的是你的所有的 zone (hint 类型的 . 除外) 的序号 (serial) 号码要跟你的数据库内容一致才行！而且不能够有出现『设定的档名:数字』的内容， 否则肯定就是配置文件有问题～上面的讯息看起来还算 OK 啦！

在上述的输出数据当中因为信息太长了，所以鸟哥将登录的时间与主机的字段拿掉了！上面是顺利启动时的状况， 如果出现问题怎办？通常出现问题的原因是因为：

- 语法设定错误： 这个问题好解决，因为在 /var/log/messages 里面有详细的说明，按照内容去修订即可；
- 逻辑设定错误： 这个就比较困扰了！为什么呢？因为他主要发生在你设定 DNS 主机的时候，考虑不周所产生的问题！例如忘记加上 (.)， 系统不会显示错误讯息，但是却会造成查询的误判，而 MX 设定的主机名错误，也不会出现有问题的讯息，但是 mail server 就是会收不到信等等～这些错误都需要很详细的 DNS client 的测试才能知道问题的所在。

我们这里先就语法设定错误方面进行介绍，至于逻辑设定的问题，那个就需要多多的进行测试才能知道了～ 底下的错误讯息都会记录在 /var/log/messages 里面喔！

```
named: /etc/named.conf:8: missing ';' before '}'
# 注意到上面提到的文件名与数字吗？说明的是 /etc/named.conf 的第 8 行，
# 至于错误是因为缺少分号 (;) 所致！去修正一下即可。

dns_rdata_fromtext: named.centos.vbird:4: near eol: unexpected end of input
zone centos.vbird/IN: loading master file named.centos.vbird: unexpected end of input
_default/centos.vbird/IN: unexpected end of input
# 指的是 named.centos.vbird 的第 4 行有问题，察看档案内容第 4 行是 SOA 的项目，
# 通常是 SOA 那五个数字没有完全！赶紧去修订一下即可啊！

dns_rdata_fromtext: named.centos.vbird:7: near 'www.centos.vbird.':
not a valid number
# 说明第 7 行在 www.centos.vbird 附近需要有一个合法的数字！刚好是 MX ，
# 所以，赶紧加上一个合法的数字，去瞧瞧改改即可！
```

通常最大的问题是...打错字！所以，务必要慢慢打字，慢慢察看清楚，尤其是登录文件内的信息喔！都处理完毕之后， 也能够透过 netstat 去查到 port 53 有在监听，再来就是要放行人家的查询了！所以，又得要修改防火墙啰！ 假设你还是安装鸟哥的防火墙脚本，那么接下来就是：

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.rule
# 找到如下两行，将批注拿掉即可！
iptables -A INPUT -p UDP -i $EXTIF --dport  53  --sport 1024:65534 -j ACCEPT
iptables -A INPUT -p TCP -i $EXTIF --dport  53  --sport 1024:65534 -j ACCEPT

[root@www ~]# /usr/local/virus/iptables/iptables.rule
```

19.4.9 步骤七：测试与数据库更新

在上面的设定都搞定，并且启动之后，你的 DNS 服务器应该是已经妥当的在运作了。 那你怎么知道你的设定是否合理？当然要作测试喔！测试有两种方式，一种是藉由 client 端的查询功能， 目的是检验你的数据库设定有无错误；另外你也可以连上底下这个网站：

- http://thednsreport.com/

这个网站可以帮你检验你的 DNS 服务器的主要设定是否有问题！不过，这个网站的检验主要是以合法授权的 zone 为主，我们自己乱搞的 DNS 是没有办法检查的啦！眞是可惜～好了，就让我们来测试测试结果吧！首先， 得将 DNS 服务器自己的 /etc/resolv.conf 改成如下模样较佳：

```
[root@www ~]# vim /etc/resolv.conf
nameserver 192.168.100.254   &lt;==自己的 IP 一定要最早出现！
nameserver 168.95.1.1
```

接下来，就让我们针对上面较重要的正、反解信息进行检测吧！同样的，鸟哥也仅列出答案的部分而已！

```
# 1\. 检查 master.centos.vbird 以及 www.centos.vbird 的 A 标志
[root@www ~]# dig master.centos.vbird
;; ANSWER SECTION:
master.centos.vbird.    600     IN      A       192.168.100.254
[root@www ~]# dig www.centos.vbird
;; ANSWER SECTION:
www.centos.vbird.       600     IN      A       192.168.100.254

# 2\. 检查 ftp.centos.vbird 与 winxp 等等的 A 标志
[root@www ~]# dig ftp.centos.vbird
;; ANSWER SECTION:
ftp.centos.vbird.       600     IN      CNAME   www.centos.vbird.
www.centos.vbird.       600     IN      A       192.168.100.254
[root@www ~]# dig winxp.centos.vbird
;; ANSWER SECTION:
winxp.centos.vbird.     600     IN      A       192.168.100.20

# 3\. 检查 centos.vbird 这个 zone 的 MX
[root@www ~]# dig -t mx centos.vbird
;; ANSWER SECTION:
centos.vbird.           600     IN      MX      10 www.centos.vbird.

# 4\. 检查 192.168.100.254 及 192.168.100.10 的反解
[root@www ~]# dig -x 192.168.100.254
;; ANSWER SECTION:
254.100.168.192.in-addr.arpa. 600 IN    PTR     www.centos.vbird.
254.100.168.192.in-addr.arpa. 600 IN    PTR     master.centos.vbird.
[root@www ~]# dig -x 192.168.100.10
;; ANSWER SECTION:
10.100.168.192.in-addr.arpa. 600 IN     PTR     slave.centos.vbird.
```

测试要成功才行呦！什么是成功呢？除了要真的有数据显示之外，该资料是否正是你要的模样？那才是顺利成功。 如果有出现错误的信息，例如找不到 www.centos.vbird 之类的，那就失败了，得要找出问题才行。

另外，如果你的数据库需要更新时，应该做哪些举动啊？举例来说，你的某个主机 IP 或者主机名要变更，也可能是新增某个主机名与 IP 的对应呢！很简单啦，通常这样做就好了：

1. 先针对要更改的那个 zone 的数据库档案去做更新，就是加入 RR 的标志即是！
2. 更改该 zone file 的序号 (Serial) ，就是那个 SOA 的第三个参数 (第一个数字)，因为这个数字会影响到 master/slave 的判定更新与否喔！
3. 重新启动 named ，或者是让 named 重新读取配置文件即可。

就这么简单啊！不过大家常常会忘记第二个步骤啦！就是将序号变大啊！如果序号没有变大，那 master/slave 的数据库可能不会主动的更新，会造成一些困扰喔！

19.5 协同工作的 DNS： Slave DNS 及子域授权设定

我们在本章一开始就曾谈过，DNS 大概是未来最重要的网络服务之一，因为所有的主机名需求都得要 DNS 提供才行。 因此，ISP 在提供 domain name 注册时，就强调得要有两部以上的 DNS 服务器才行。而为了简化 DNS 管理人员的负担， 使用 Master/Slave DNS 架构的情况会比较好！为什么呢？让我们再回忆一下 Slave DNS 的特色：

- 为了不间断的提供 DNS 服务，你的领域至少需要有两部 DNS 服务器来提供查询的功能；
- 承上，这几部 DNS 服务器应该要分散在两个以上的不同 IP 网域才好；
- 为方便管理，通常除了一部主要 Master DNS 之外，其他的 DNS 会使用 slave 的模式；
- slave DNS 服务器本身并没有数据库，他的数据库是由 master DNS 所提供的；
- master/slave DNS 必需要可以相互传输 zone file 的相关信息才行，这部份需要 /etc/named.conf 之设定辅助。

除此之外，如果你有朋友或者是学生想要跟你要一个子域，那又该如何设定另一部 DNS 服务器呢？就让我们依序来谈谈啰～

19.5.1 master DNS 权限的开放

我们使用 19.4.3 的案例，继续来架设一部支持该案例的 slave DNS 吧！基本的假设为：

- 提供 slave DNS 服务器进行 zone transfer 的服务器为 master.centos.vbird
- centos.vbird 及 100.168.192.in-addr.arpa 两个 zone 都提供给 slave DNS 使用
- master.centos.vbird 的 named 仅提供给 slave.centos.vbird 这部主机进行 zone transfer
- Slave DNS server 架设在 192.168.100.10 这部服务器上面 (所以 zone file 要修订)

如上所示，我们的 master.centos.vbird 这部服务器除了 named.conf 需要调整之外，两个 zone file 也都需要调整！ 在 named.conf 当中，需要设定哪个 IP 可以对我的 zone 进行传输 (allow-transfer)，而在 zone file 当中，就是各加入一笔 NS 的记录即可！增加的部分如下所示：

```
# 1\. 修订 named.conf，主要修改 zone 参数内的 allow-transfer 项目
[root@www ~]# vim /etc/named.conf
....前面省略....
zone "centos.vbird" IN {
        type master;
        file "named.centos.vbird";
 allow-transfer { 192.168.100.10; };  // 在这里新增 slave 的 IP
};
zone "100.168.192.in-addr.arpa" IN {
        type master;
        file "named.192.168.100";
 allow-transfer { 192.168.100.10; };  // 在这里新增 slave 的 IP
};
```

在上头所列示的那两个数据库档案当中，你必须要新增所需要的 NS 标志才行！NS 对应的主机名为 slave.centos.vbird， IP 则是 192.168.100.10 呦！结果如下：

```
# 2\. 在 zone file 里面新增 NS 标志，要注意需要有 A(正解) 及 PTR(反解) 的设定
[root@www ~]# vim /var/named/named.centos.vbird
$TTL    600
@                        IN SOA   master.centos.vbird. vbird.www.centos.vbird. (
                                  2011080402 3H 15M 1W 1D )
@                        IN NS    master.centos.vbird.
@                        IN NS    slave.centos.vbird.
master.centos.vbird.     IN A     192.168.100.254
slave.centos.vbird.      IN A     192.168.100.10
@                        IN MX 10 www.centos.vbird.
....(底下省略)....

[root@www ~]# vim /var/named/named.192.168.100
$TTL    600
@       IN SOA  master.centos.vbird. vbird.www.centos.vbird. (
                2011080402 3H 15M 1W 1D )
@       IN NS   master.centos.vbird.
@       IN NS   slave.centos.vbird.
254     IN PTR  master.centos.vbird.
10      IN PTR  slave.centos.vbird.
....(底下省略)....
# 要特别注意一件事，那就是，你的 zone file 内的序号要增加！鸟哥测试日期是 8/4，
# 第 2 次进行，所以序号就以该天的日期为准来设计的！最后记得 restart 一下啦！

[root@www ~]# /etc/init.d/named restart
[root@www ~]# tail -n 30 /var/log/messages &#124; grep named
starting BIND 9.7.0-P2-RedHat-9.7.0-5.P2.el6_0.1 -u named -t /var/named/chroot
....(中间省略)....
zone 100.168.192.in-addr.arpa/IN: loaded serial 2011080402
zone centos.vbird/IN: loaded serial 2011080402
zone 100.168.192.in-addr.arpa/IN: sending notifies (serial 2011080402)
zone centos.vbird/IN: sending notifies (serial 2011080402)
```

反正重新启动过 named 后，直觉记得就是要查阅 messages 登录信息就对了。从上表的输出来看，会多一个 sending notifies (传送注意事项) 关键词的数据，那就是提醒 slave DNS 来比对序号大小了！所以，你说，序号有没有很重要呢？当然很重要啊！ 连登录讯息都会告知序号的大小哩！这样 master DNS 就设定妥当啰！接下来玩玩 Slave 的设定吧！

19.5.2 Slave DNS 的设定与数据库权限问题

既然 Slave DNS 也是 DNS 服务器嘛！所以，当然也是需要安装 bind, bind-chroot 等等的软件！ 这部份回去 19.3.1 里面瞧瞧即可，反正记得使用 yum 安装就对了。 接下来得要设定 named.conf 吧？而既然 Master/Slave 的数据库是相同的，所以，理论上， named.conf 内容就是大同小异啰～ 唯一要注意的就是 zone type 类型的差异，以及宣告 master 在哪里就是了。 至于 zone filename 部分，由于 zone file 都是从 master 取得的，透过 named 这个程序来主动建立起需要的 zone file，因此这个 zone file 放置的目录权限就很重要！让我们直接来处理看看：

```
# 1\. 准备 named.conf 的内容：
[root@clientlinux ~]# vim /etc/named.conf
....(前面的部分完全与 master.centos.vbird 相同，故省略)....
zone "centos.vbird" IN {
        type slave;
        file "slaves/named.centos.vbird";
        masters { 192.168.100.254; };
};
zone "100.168.192.in-addr.arpa" IN {
        type slave;
        file "slaves/named.192.168.100";
        masters { 192.168.100.254; };
};

# 2\. 检查 zone file 预计建立的目录权限是否正确！底下目录为系统默认值：
[root@clientlinux ~]# ll -d /var/named/slaves
drwxrwx---. 2 named named 4096 2011-06-25 11:48 /var/named/slaves
# 注意权限、使用者以及群组三个字段的数据！需要与 named 这个用户及群组有关！

[root@clientlinux ~]# ll -dZ /var/named/slaves
drwxrwx---. named named system_u:object_r:named_cache_t:s0 /var/named/slaves
# 也不要忘记与 SELinux 有关的事情！
```

为了方便使用者设定，CentOS 预设在 /var/named/slaves/ 处理好了相关权限～所以你可以轻松的处理权限问题～ 我们就建议你的 slave zone file 放置在该目录下！所以上表当中的 file 参数才会这么写～此外，那个 masters 结尾有个 s 喔！这里最容易写错～那么要不要处理 zone file 呢？除了 named.ca 这个 . 需要主动存在之外， 另外两个 type slave 的数据库档案，当然不必存在啊！因为会从 master 处取得嘛！接下来，就让我们来启动 named 并进行观察吧！

```
[root@clientlinux ~]# /etc/init.d/named start
[root@clientlinux ~]# chkconfig named on
[root@clientlinux ~]# tail -n 30 /var/log/messages &#124; grep named
starting BIND 9.7.0-P2-RedHat-9.7.0-5.P2.el6_0.1 -u named -t /var/named/chroot
loading configuration from '/etc/named.conf'
....(中间省略)....
running
zone 100.168.192.in-addr.arpa/IN: Transfer started.
zone 100.168.192.in-addr.arpa/IN: transferred serial 2011080402
zone centos.vbird/IN: Transfer started.
zone centos.vbird/IN: transferred serial 2011080402  &lt;==注意序号正确否
# 你会看到如上的讯息，重点是还有告知序号喔！非常重要！

[root@clientlinux ~]# ll /var/named/slaves
-rw-r--r--. 1 named named 3707 2011-08-05 14:12 named.192.168.100
-rw-r--r--. 1 named named  605 2011-08-05 14:12 named.centos.vbird
# 这两个 zone file 会主动被建立起来呢！

[root@clientlinux ~]# dig master.centos.vbird @127.0.0.1
[root@clientlinux ~]# dig -x 192.168.100.254 @127.0.0.1
# 上述两个检测的指令如果是正确的显示出 A 与 PTR 的话，那就完成了！
```

你瞧！如此一来你的 zone file 就会主动的被建立起来喔！未来如果你的 master DNS 要更新数据库时， 只要修改过序号，并重新启动 named 后，这部 slave DNS 就会跟着更新啦！ 啊！眞是『福气啦！』！！ 不过，如果你发现到启动 slave DNS 时，你的登录信息竟然是这样：

```
zone centos.vbird/IN: Transfer started.
transfer of 'centos.vbird/IN' from 192.168.100.254#53: connected using
192.168.100.10#58187
dumping master file: tmp-a1bYfCd3i3: open: permission denied
transfer of 'centos.vbird/IN' from 192.168.100.254#53: failed while receiving
responses: permission denied
transfer of 'centos.vbird/IN' from 192.168.100.254#53: end of transfer
```

如果出现类似这样的讯息时，不必怀疑啦！肯定是权限错误啦！请再次检查你的数据库档案所放置的目录权限是否可以让 named 写入啊！处理处理就好了！现在，你的 DNS 会变的更加强壮啰！因为有类似的备援系统啰～不过仍然要注意的是， 网络查询 centos.vbird 时，master 与 slave 的地位是相同的，并不是 master 挂点才使用 slave 来查询喔！所以，这两部服务器的相同 domain 的数据库内容要完全一致才行！

19.5.3 建置子域 DNS 服务器：子域授权课题

除了 Master/Slave 需要协同 DNS 服务器共同提供服务之外，DNS 之间如果有上层、下属的关系时，该如何设定？ 亦即，假设我的网域很大，我只想要负责上层的 DNS 而已，下层希望直接交给各单位的负责人来负责，要怎么设定呢？ 举个例子来说，以成大为例，成大计中仅管理各个系所的 DNS 服务器 IP 而已，由于各个系所的主机数量可能很大， 如果每个人都要请计中来设定，那么管理员可能会疯掉，而且在实际设计上也不太人性化。

所以啰，计中就将各个 subdomain (子域) 的管理权交给各个系所的主机管理员去管理，如此一来， 各系所的设定上面会比较灵活，且上层 DNS 服务器管理员也不用太麻烦呐！

好了，那么如何开放子域授权呢？我们以刚刚在 master 上面建立的 centos.vbird 这个 zone 为例， 假设今天你是个 ISP ，有个人想要跟妳申请 domain name ，他要的 domain 是『 niki.centos.vbird 』， 那你该如何处理？

- 上层 DNS 服务器：亦即是 master.centos.vbird 这一部，只要在 centos.vbird 那个 zone file 内，增加指定 NS 并指向下层 DNS 的主机名与 IP (A) 即可，而 zone file 的序号也要增加才行；
- 下层 DNS 服务器：申请的领域名必须是上层 DNS 所可以提供的名称，并告知上层 DNS 管理员，我们这个 zone 所需指定的 DNS 主机名与对应的 IP 即可。然后就开始设定自己的 zone 与 zone file 相关数据。

假设我们管理 niki.centos.vbird 的服务器主机名为 dns.niki.centos.vbird ，而这部主机的 IP 为 192.168.100.200， 那接下来就让我们实际来设定吧！

- 上层 DNS 服务器：只需新增 zone file 的 NS 与 A 即可

上层 DNS 的处理真是简单到爆炸！我们只要修改 master DNS (www.centos.vbird 那一部) 里面的 named.centos.vbird 这个正解档案即可。slave DNS 不用修改，是因为他会自动更新嘛！新增如下的数据即可：

```
[root@www ~]# vim /var/named/named.centos.vbird
@                   IN SOA   master.centos.vbird. vbird.www.centos.vbird. (
                             2011080501 3H 15M 1W 1D )
# 上面的 SOA 部分序号加大，底下新增这两行即可 (原本的数据都保留不动)！
niki.centos.vbird.      IN NS    dns.niki.centos.vbird.
dns.niki.centos.vbird.  IN A     192.168.100.200

[root@www ~]# /etc/init.d/named restart
[root@www ~]# tail -n 30 /var/log/messages &#124; grep named
Aug  5 14:22:36 www named[9564]: zone centos.vbird/IN: loaded serial 2011080501
# 登录档的关键是上面的序号部分～必须是我们填写的新的序号才对！

[root@www ~]# dig dns.niki.centos.vbird @127.0.0.1
# 你会发现是错误的！找不到 A 喔！
```

上层 DNS 的设定非常简单！只要修改 zone file 即可～不过，由于 zone file 指定的是 NS 的查询权功能， 因此，最后那个指令在 dig dns.niki.centos.vbird 时，却会找不到 A 喔！那是正常的～因为 192.168.100.200 尚未设定好 niki.centos.vbird 这个领域嘛！所以追踪的结果并没有发现在 192.168.100.200 有 niki.centos.vbird 的 zone 啊！ 所以当然找不到。此时数据库的管理权在 192.168.100.200 上啦！这样可以理解吗？再来处理下层 DNS 吧！

- 下层 DNS 服务器：需要有完整的 zone 相关设定

下层的 DNS 设定就与 19.4 的详细内容一样了！所以在这里我们仅列出重要的项目：

```
# 1\. 修改 named.conf ，增加 zone 的参数，假设档名为 named.niki.centos.vbird
[root@niki ~]# vim /etc/named.conf
....(前面省略)....
zone "niki.centos.vbird" IN {
        type master;
        file "named.niki.centos.vbird";
};

# 2\. 建立 named.niki.centos.vbird
[root@niki ~]# vim /var/named/named.niki.centos.vbird
$TTL   600
@      IN SOA   dns.niki.centos.vbird. root.niki.centos.vbird. (
                2011080501 3H 15M 1W 1D )
@      IN NS    dns.niki.centos.vbird.
dns    IN A     192.168.100.200
www    IN A     192.168.100.200
@      IN MX 10 www.niki.centos.vbird.
@      IN A     192.168.100.200
# 为了简化整个版面，所以鸟哥都使用 hostname 而非 FQDN！请见谅！

# 3\. 启动并观察相关登录信息
[root@niki ~]# /etc/init.d/named restart
[root@niki ~]# tail -n 30 /var/log/messages &#124; grep named
....(前面省略)....
zone niki.centos.vbird/IN: loaded serial 2011080501
....(底下省略)....
# 同时，&lt;u&gt;记得处理一下防火墙的放行问题&lt;/u&gt;！否则测试会失败！！

[root@niki ~]# dig www.niki.centos.vbird @192.168.100.254
# 上述的动作必须要有响应才行！否则就会出问题～
```

19.5.4 依不同接口给予不同的 DNS 主机名： view 功能的应用

想象一个环境，以我们目前的局域网络服务器来说，我的 master.centos.vbird 有两个界面，分别是 192.168.100.254/24 (对内) 及 192.168.1.100/24 (对外)，那当我外边的用户想要了解到 master.centos.vbird 这部服务器的 IP 时，取得的竟然是 192.168.100.254，因此还得要透过 NAT 才能联机到该接口，但明明 192.168.100.254 与外部的 192.168.1.100 是同一台服务器主机嘛！干嘛还得要经过 NAT 转到内部接口呢？有没有办法让外部的查询找到 master.centos.vbird 是 192.168.1.100 而内部的找到则回应 192.168.100.254 呢？可以的！那就透过 view 的功能！

那么 view 要怎么处理呢？其实就是让不同来源的用户，能够取得他们自己的 zone 响应就是了。举例来说，当用户来自 10.0.0.1 时，这个来源不可能是内部 (192.168.100.0/24) ，因此这个来源就会使用外部的 zone file 内容来响应。 因此，我们就得要准备同一个 zone 需要两个不同的设定，再将个别的设定带入自己的客户端查询当中。

现在我们针对这个概念，对于鸟哥的区网设定 view 的原则是这样的：

- 建立一个名为 intranet 的名字，这个名字代表客户端为 192.168.100.0/24 的来源；
- 建立一个名为 internet 的名字，这个名字代表客户端为非 192.168.100.0/24 的其他来源
- intranet 使用的 zone file 为本章前面各小节所建立的 zone filename，internet 使用的 zone filename 则在原本的档名后面累加 inter 的扩展名，并修订各标志的结果。

再次强调，最终的结果当中，从内网查到的 www.centos.vbird IP 应该是 192.168.100.254，而只要不是鸟哥内网来源的客户端， 查到的 www.centos.vbird IP 应该是 192.168.1.100 才对！那就让我们来实际设定此一项目吧！

```
[root@www ~]# vim /etc/named.conf
options {
        directory       "/var/named";
        dump-file       "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
        memstatistics-file "/var/named/data/named_mem_stats.txt";
        allow-query     { any; };
        recursion yes;
        allow-transfer  { none; };
};

acl intranet { 192.168.100.0/24; };        &lt;==针对 intranet 给予的来源 IP 指定
acl internet { ! 192.168.100.0/24; any; }; &lt;==加上惊叹号 (!) 代表反向选择的意思

view "lan" {                              &lt;==只是一个名字，代表的是内网
        match-clients { "intranet"; };  &lt;==吻合这个来源的才使用底下的 zone
        zone "." IN {
                type hint;
                file "named.ca";
        };
        zone "centos.vbird" IN {
                type master;
                file "named.centos.vbird";
                allow-transfer { 192.168.100.10; };
        };
        zone "100.168.192.in-addr.arpa" IN {
                type master;
                file "named.192.168.100";
                allow-transfer { 192.168.100.10; };
        };
};

view "wan" {                             &lt;==同样，只是个名字而已！
        match-clients { "internet"; }; &lt;==代表的则是外网的 internet 来源
        zone "." IN {
                type hint;
                file "named.ca";
        };
        zone "centos.vbird" IN {
                type master;
                file "named.centos.vbird.inter"; &lt;==档名必须与原有的不同！
        };
        // 外网因为没有使用到内网的 IP，所以 IP 反解部分可以不写于此
};
```

上表中，有些数据是重复的，有些则需要经过修改。现在，让我们来改改 named.centos.vbird.inter 吧！

```
[root@www ~]# cd /var/named
[root@www named]# cp -a named.centos.vbird named.centos.vbird.inter
[root@www named]# vim named.centos.vbird.inter
$TTL    600
@                        IN SOA   master.centos.vbird. vbird.www.centos.vbird. (
                                  2011080503 3H 15M 1W 1D )
@                        IN NS    master.centos.vbird.
master.centos.vbird.     IN A     192.168.1.100
@                        IN MX 10 www.centos.vbird.

www.centos.vbird.        IN A     192.168.1.100
linux.centos.vbird.      IN CNAME www.centos.vbird.
ftp.centos.vbird.        IN CNAME www.centos.vbird.
forum.centos.vbird.      IN CNAME www.centos.vbird.
workstation.centos.vbird. IN A    192.168.1.101

[root@www named]# /etc/init.d/named restart
[root@www named]# tail -n 30 /var/log/messages
[root@www named]# dig www.centos.vbird @192.168.100.254
www.centos.vbird.        600     IN      A       192.168.100.254
# 要得到上面的 IP 才是对的喔！因为接口来自于 192.168.100.0/24 网段

[root@wwww named]# dig www.centos.vbird @192.168.1.100
www.centos.vbird.        600     IN      A       192.168.1.100
# 要得到上面的 IP 才是对的喔！因为接口来自非 192.168.100.0/24 网段
```

有没有很简单！这样就能让你的 DNS 依据不同的用户来源，分别给予同一个主机名的不同解析呢！

例题：你的网站读者非常的多，但是分布在世界各地。你想让亚洲区的读者联机到台湾的站台，而其他国家的联机则连到美国的站台， 但又不想要让使用者自己挑选不同的主机名，想使用同一组主机名，此时该如何是好？答：鸟哥可以想到的最简单的方案，就是透过 DNS 来设定相同主机名的不同 IP 目标，亦即是透过 view 来规范即可。 不过，与上述鸟哥的区网简单范例不同，我们得要收集亚洲区的 IP 才行，这些区段可能可以透过底下的网站来取得：

- 五大洲的 IP 管理所属人：http://www.iana.org/numbers/
- 每个单位的 IP 分布： http://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xml
- 台湾地区 IP 分布： http://rms.twnic.net.tw/twnic/User/Member/Search/main7.jsp?Order=inet_aton%28Startip%29

然后再透过 acl 以及 view 来规范即可。鸟哥的收集资料如下，如果有误，还请告知！

```
acl asia { 1.0.0.0/8;  14.0.0.0/8;  27.0.0.0/8;  36.0.0.0/8;  39.0.0.0/8;
          42.0.0.0/0;  49.0.0.0/8;  58.0.0.0/8;  59.0.0.0/8;  60.0.0.0/8;
          61.0.0.0/8; 101.0.0.0/8; 103.0.0.0/8; 106.0.0.0/8; 110.0.0.0/8;
         111.0.0.0/8; 112.0.0.0/8; 113.0.0.0/8; 114.0.0.0/8; 115.0.0.0/8;
         116.0.0.0/8; 117.0.0.0/8; 118.0.0.0/8; 119.0.0.0/8; 120.0.0.0/8;
         121.0.0.0/8; 122.0.0.0/8; 123.0.0.0/8; 124.0.0.0/8; 125.0.0.0/8;
         126.0.0.0/8; 175.0.0.0/8; 180.0.0.0/8; 182.0.0.0/8; 183.0.0.0/8;
         202.0.0.0/8; 203.0.0.0/8; 210.0.0.0/8; 211.0.0.0/8; 218.0.0.0/8;
         219.0.0.0/8; 220.0.0.0/8; 221.0.0.0/8; 222.0.0.0/8; 223.0.0.0/8;
         139.175.0.0/16; 140.0.0.0/8;150.116.0.0/16;150.117.0.0/16;
         163.0.0.0/8; 168.95.0.0/16;192.0.0.0/8;
};
acl nonasia { ! "asia"; any; };
```

如上所示，加入 asia 与 nonasia 的相关设定，再使用 view 来处理相关的 zone ，并修改 zone file 内容， 就能够处理好这个案例的需求啰！

19.6 DNS 服务器的进阶设定

其实， DNS 服务器的运作原理与架设方式的变化，眞的很高深莫测的！在这里，我们额外的提出一些比较进阶的内容给大家参考参考， 例如架设一个合法授权的 DNS 服务器以及利用 rndc 控管 DNS 系统喔！

19.6.1 架设一个合法授权的 DNS 服务器

好啦！现在你应该知道什么是『经上游授权的合法 DNS 服务器』了吧？ 没错！就是上游的 DNS 服务器将子域的查核权开放给你来设定就对啦！嗯！虽然知道原理，但是那么我要如何来架设一个合法的 DNS 服务器呢？好让我自己管理自己的 domain！举例来说，鸟哥的 vbird.idv.tw 就是鸟哥自己管理的哩～底下我们就来谈一谈，如何向 ISP 申请一个合法授权的 DNS 服务器，或者是合法的主机名啊！

- 1. 申请一个合法的 domain name ...就是要花钱！

既然是要建立一个合法的 DNS server，自然就要向合法的 ISP 申请授权啰！目前你可以到底下的地方去申请喔！

- http://www.twnic.net/index3.php

其实 TWNIC 已经将台湾地区的一些 domain 授权给各大 ISP 管理了，所以你连接上述的网站之后，可以点选里头相关的连结到各大 ISP 去注册！例如鸟哥就在 Hinet 注册了 vbird.idv.tw 这个网域！现在鸟哥就以 Hinet 的注册做为说明吧：

1. 进入主画面： 直接连结到底下的网页去： http://domain.hinet.net
2. 选择需要的域名，并查询该网域是否已存在： 因为网域必需是独一无二的，所以你必需使用该网页当中提供的查询功能， 去查询一下你想要的网域是否已经被注册了呢？一定要没有被注册的网域才可以喔！
3. 逐步进行注册： 你可以选择很多种类的领域来注册，如果想要注册个人网站，请按下图所指的 (1) 处，如果想要注册类似 vbird.tw 这种网域的话，则可以选择 (2) 所指的那个项目。然后以该网站提供的功能一步一步的往下去进行， 例如以鸟哥的『个人网址』之注册为例，按下个人网址之后，会出现流程步骤为：

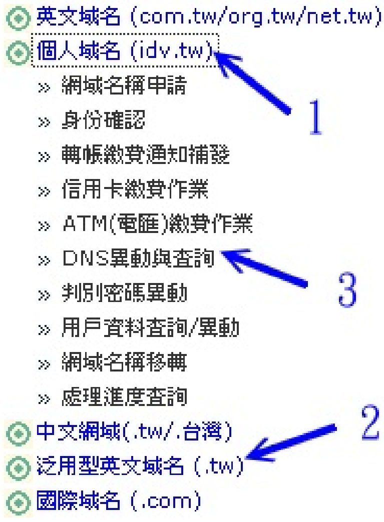


图 19.6-1、以 Hinet 网站为依据介绍注册 domain 的方法

请依序一步一步的将他完成，最后你会得到一组账号密码，就能够修改自己的领域啦！

4. 选择网站代管或架设 DNS 模式： 我们可以直接请 ISP 帮我们设定好 host 对应 IP 就好(最多三部)，当然也可以自行设定一下我们所需要的 DNS 服务器啦！如果未来你可能会架设 mail server ，所以还是自行设定 DNS 主机好了！你可以选择图 19.6-1 在 (3) 所指的『DNS异动与查询』项目，会出现下面图标。记得选择『DNS』及填写你的 hostname 与正确的 IP 即可喔！注意： 要填选这个项目，最好你的 IP 是固定制的，浮动制的 IP 不建议用这个选项！



图 19.6-2、以 Hinet 网站为依据介绍注册 domain 的方法

5. 2. 以 DNS 服务器的详细设定 (19.4) 之设定内容来设定你的主机：

如果你已经以 DNS 服务器的方式申请了一个 domain name ，那么你就必须要设定你的 DNS 主机了！ 请注意，这个情况之下，你只要设定你的注册的网域的正解即可！ 反解部分则先不要理会，当然，如果你有办法的话，最好还是请上层的 ISP 帮你设定啰！

- 3. 测试：

设定一部合法的 DNS 完毕后，建议你可以到这个网站去查询一下你的设定是否妥当：

- http://thednsreport.com/

如此一来，你的 DNS 主机上面设定的任何信息，都可以透过 Internet 上面的任何一部主机来查询到喔！够棒吧！心动了吗？赶快去试看看吧！ ^_^

19.6.2 LAME Server 的问题

或许你曾经在 /var/log/messages 里面看到类似这样的讯息：

```
[root@www ~]# more /var/log/messages
1 Oct  5 05:02:30 test named[432]: lame server resolving '68.206.244.205.
  in-addr.arpa' (in '206.244.205.in-addr.arpa'?): 205.244.200.3#53
2 Oct  5 05:02:31 test named[432]: lame server resolving '68.206.244.205.
  in-addr.arpa' (in '206.244.205.in-addr.arpa'?): 206.105.201.35#53
3 Oct  5 05:02:41 test named[432]: lame server resolving '68.206.244.205.
  in-addr.arpa' (in '206.244.205.in-addr.arpa'?): 205.244.112.20#53
```

这是什么东西呐？根据官方提供的文件资料来看 (在你的 CentOS 6.x 的系统下，请察看这个档案『 /usr/share/doc/bind-9.7.0/arm/Bv9ARM.ch06.html 』)，当我们的 DNS 服务器在向外面的 DNS 系统查询某些正反解时，可能由于『对方』 DNS 主机的设定错误，导致无法解析到预期的正反解结果，这个时候就会发生所谓的 lame server 的错误！

那么这个错误会让我们的 DNS 服务器发生什么严重的后果吗？既然仅是对方的设定错误，所以自然就不会影响我们的 DNS 服务器的正常作业了。 只是我们的 DNS 主机在查询时，会发生无法正确解析的警告讯息而已， 这个讯息虽然不会对我们的 Linux 主机发生什么困扰，不过，对于系统管理员来说， 要天天查询的 /var/log/messages 档案竟然有这么多的登录信息，这是很讨厌的一件事！

好了，我们知道 lame server 是对方主机的问题，对我们主机没有影响，但是却又不想要让该讯息出现在我们的登录档 /var/log/messages 当中，怎么达到这样的功能呢？呵呵！就直接利用 BIND 这个软件所提供的登录文件参数啊！ 动作很简单，在你的 /etc/named.conf 档案当中的最底下，加入这个参数即可：

```
# 1\. 修改 /etc/named.conf
[root@www ~]# vim /etc/named.conf
// 加入底下这个参数：
logging {
        category lame-servers { null; };
};

# 2\. 重新启动 bind
[root@www ~]# /etc/init.d/named restart
```

基本上，那个 logging 是主机的登录文件记录的一个设定项目，因为我们不要 lame server 的信息， 所以才将他设定为无 (null) ，这样就改完了！记得重新启动 named 之后，还是要察看一下 /var/log/messages 喔！ 以确定 named 的正确启动与否！然后，嘿嘿，以后就不会看到 lame server 咯！

19.6.3 利用 RNDC 指令管理 DNS 服务器

不知道你会不会觉得很奇怪，那就是为啥启动 DNS 后，在 /var/log/messages 老是看到这一句话：

```
command channel listening on 127.0.0.1#953
```

而且在本机端的 port 953 还多了个 named 所启动的服务，那是啥？那就是所谓的 rndc 了。这个 rndc 是 BIND version 9 以后所提供的功能啦，他可以让你很轻松的管理你自己的 DNS 服务器喔！ 包括可以检查已经存在 DNS 快取当中的资料、重新更新某个 zone 而不需要重新启动整个 DNS ， 以及检查 DNS 的状态与统计资料等等的，挺有趣的！

不过，因为 rndc 可以很深入的管理你的 DNS 服务器，所以当然要进行一些控管啦！ 控管的方式是经过 rndc 的设定来建立一支密钥 (rndc key)，并将这支密钥相关的信息写入你的 named.conf 配置文件当中，重新启动 DNS 后，你的 DNS 就能够藉由 rndc 这个指令来管理啰！ 事实上，新版的 distributions 通常已经帮你主动的建立好 rndc key 了，所以你不需要忙碌～ 不过，如果你还是在登录档当中发现一些错误，例如：

```
couldn't add command channel 127.0.0.1#953: not found
```

那就表示你 DNS 的 rndc key 没有设定好啦！那要如何设定好？很简单～只要先建立一把 rndc key ，然后加到 named.conf 当中去即可！你可以使用 bind 提供的指令来进行这样的工作喔！

```
# 1\. 先建立 rndc key 的相关数据吧！
[root@www ~]# rndc-confgen
# Start of rndc.conf &lt;==底下没有 # 的第一部份请复制到 /etc/rndc.conf 中
key "rndc-key" {
        algorithm hmac-md5;
        secret "UUqxyIwui+22CobCYFj5kg==";
};

options {
        default-key "rndc-key";
        default-server 127.0.0.1;
        default-port 953;
};
# End of rndc.conf

# 至于底下的 key 与 controls 部分，则请复制到 named.conf 且解开 # 喔！
# Use with the following in named.conf, adjusting the allow list as needed:
# key "rndc-key" {
#       algorithm hmac-md5;
#       secret "UUqxyIwui+22CobCYFj5kg==";
# };
#
# controls {
#       inet 127.0.0.1 port 953
#               allow { 127.0.0.1; } keys { "rndc-key"; };
# };
# End of named.conf
# 请注意，这个 rndc-confgen 是利用随机数计算出加密的那把 key ，
# 所以每次执行的结果都不一样。所以上述的数据与你的屏幕会有点不同。

# 2\. 建立 rndc.key 档案
[root@www ~]# vim /etc/rndc.key
# 在这个档案当中将原本的数据全部删除，并将刚刚得到的结果给他贴上去
key "rndc-key" {
        algorithm hmac-md5;
        secret "UUqxyIwui+22CobCYFj5kg==";
};

# 3\. 修改 named.conf
[root@www ~]# vim /etc/named.conf
# 在某个不被影响的角落建置如下的内容：
key "rndc-key" {
       algorithm hmac-md5;
       secret "UUqxyIwui+22CobCYFj5kg==";
};
controls {
       inet 127.0.0.1 port 953
               allow { 127.0.0.1; } keys { "rndc-key"; };
};

[root@www ~]# /etc/init.d/named restart
```

建立了rndc key 并且启动 DNS ，同时你的系统也已经有 port 953 之后，我们就可以在本机执行 rndc 这个指令了。这个指令的用法请直接输入 rndc 来查询即可：

```
[root@www ~]# rndc
Usage: rndc [-c config] [-s server] [-p port]
        [-k key-file ] [-y key] [-V] command

command is one of the following:

  reload        Reload configuration file and zones.
  stats         Write server statistics to the statistics file.
  dumpdb        Dump cache(s) to the dump file (named_dump.db).
  flush         Flushes all of the server's caches.
  status        Display status of the server.
# 其他就给他省略啦！请自行输入这个指令来参考啰！
```

那如何使用呢？我们举几个小例子来说明吧！

```
# 范例一：将目前 DNS 服务器的状态显示出来
[root@www ~]# rndc status
version: 9.7.0-P2-RedHat-9.7.0-5.P2.el6_0.1
CPUs found: 1
worker threads: 1
number of zones: 27         &lt;==这部 DNS 管理的 zone 数量
debug level: 0              &lt;==是否具有 debug 及 debug 的等级
xfers running: 0
xfers deferred: 0
soa queries in progress: 0
query logging is OFF        &lt;==是否具有 debug 及 debug 的等级
recursive clients: 0/0/1000
tcp clients: 0/100
server is up and running    &lt;==是否具有 debug 及 debug 的等级

# 范例二：将目前系统的 DNS 统计数据记录下来
[root@www ~]# rndc stats
# 此时，预设会在 /var/named/data 内产生新档案，你可以去查阅：
[root@www ~]# cat /var/named/data/named_stats.txt
+++ Statistics Dump +++ (1312528012)
....(中间省略)....
++ Zone Maintenance Statistics ++
                   2 IPv4 notifies sent
++ Resolver Statistics ++
....(中间省略)....
++ Cache DB RRsets ++
[View: lan (Cache: lan)]
[View: wan (Cache: wan)]
[View: _bind (Cache: _bind)]
[View: _meta (Cache: _meta)]
++ Socket I/O Statistics ++
                   5 UDP/IPv4 sockets opened
                   4 TCP/IPv4 sockets opened
                   2 UDP/IPv4 sockets closed
                   1 TCP/IPv4 sockets closed
                   2 TCP/IPv4 connections accepted
++ Per Zone Query Statistics ++
--- Statistics Dump --- (1312528012)

# 范例三：将目前高速缓存当中的数据记录下来
[root@www ~]# rndc dumpdb
# 与 stats 类似，会将 cache 的数据放置成为一个档案，你可以去查阅：
# /var/named/data/cache_dump.db
```

如果你在执行 rndc 指令时老是出现如下错误：

```
rndc: connection to remote host closed
This may indicate that the remote server is using an older version of
the command protocol, this host is not authorized to connect,
or the key is invalid.
```

这表示你的 /etc/rndc.key 与 /etc/rndc.conf 内密钥的编码不同所致。 请你自行以上述的 rndc-confgen 的方式自行处理你的 rndc key ，并重新启动 named 即可啊！ 用这东西管理，你就不需要每次都重新启动 named 啰！ ^_^

19.6.4 架设动态 DNS 服务器： 让你成为 ISP 啦！

什么是动态 DNS (Dynamic DNS, DDNS) 主机呢？还记得我们在第十章里面提到， 如果我们本身是以拨接制的 ADSL 连上 Internet 时，我们的 IP 通常是 ISP 随机提供的，因此每次上网的 IP 都不固定，所以， 我们没有办法以上面的 DNS 设定来给予这种连上 Internet 的方法一个适当的主机名。

也因此，如果我们想要利用这种没有固定 IP 的联机方法架设网站时，就得要有特殊的管道了～ 其中之一的方法就是利用 Internet 上面已经提供的免费动态 IP 对应主机名的服务！ 例如： http://www.no-ip.org 。

提供这样的服务利用的是什么原理呢？基本上， DNS 主机还是得要提供 Internet 相关的 zone 的主机名与 IP 的对应数据才行，所以，DDNS 主机 就必须要提供一个机制，让客户端可以透过这个机制来修改他们在 DDNS 主机上面的 zone file 内的资料才行。

那会不会很难啊？不会啊！我们的 BIND 9 就有提供类似的机制啦！那就是利用 update-policy 这个选项，配合认证用的 key 来进行数据文件的更新。简单的说， 1) 我们的 DDNS 主机先提供 Client 一把 Key (就是认证用的数据， 你可以将他想成是账号与密码的概念)， 2) Client 端利用这把 Key ，并配合 BIND 9 的 nsupdate 指令， 就可以连上 DDNS 主机，并且修改主机上面的 Zone file 内的对应表了。感觉上很像很简单喔！ 没错啊！架设上真的很简单的～底下我们就来尝试设定一下喔：

- 1. DDNS Server 端的设定：

假设我有一个朋友，他使用的 Linux 主机的 IP 是会随时变动的，但是他想要架设 Web 网站， 所以他向我申请了一个领域名，那就是 web.centos.vbird ，此时我必需要给他一把密钥， 并且设定我的 named.conf 让 centos.vbird 这个 zone 能够接受来自客户端的数据更新才行！首先来建立这把密钥吧！

```
[root@www ~]# dnssec-keygen -a [算法] -b [密码长度] -n [类型] 名称
选项与参数：
-a ：后面接的 [type] 为演算方式的意思，主要有 RSAMD5, RSA, DSA, DH
     与 HMAC-MD5 等。建议你可以使用常见的 HMAC-MD5 来演算密码；
-b ：你的密码长度为多少？通常给予 512 位的 HMAC-MD5；
-n ：后面接的则是客户端能够更新的类型，主要有底下两种，建议给 HOST 即可：
     ZONE：客户端可以更新任何标志及整个 ZONE；
     HOST：客户端仅可以针对他的主机名来更新。

[root@www ~]# cd /etc/named
[root@www named]# dnssec-keygen -a HMAC-MD5 -b 512 -n HOST web
Kweb.+157+36124
[root@www named]# ls -l
-rw-------. 1 root root 112 Aug  5 15:22 Kweb.+157+36124.key
-rw-------. 1 root root 229 Aug  5 15:22 Kweb.+157+36124.private
# 上面那把是公钥，下面那把则是私钥档案！

[root@www named]# cat Kweb.+157+36124.key  &lt;==看一下公钥！
web. IN KEY 512 3 157 xZmUo8ozG8f2OSg/cqH8Bqxk59Ho8....3s9IjUxpFB4Q==
# 注意到最右边的那个密码长度，等一下我们要复制的仅有那个地方！
```

接下来你必需要：将公钥的密码复制到 /etc/named.conf 当中，将私钥传给你的 web.centos.vbird 那部主机上！好了，那就开始来修改 named.conf 内的相关设定吧！

```
[root@www ~]# vim /etc/named.conf
// 先在任意地方加入这个 Key 的相关密码信息！
key "web" {
        algorithm hmac-md5;
        secret "xZmUo8ozG8f2OSg/cqH8Bqxk59Ho8....3s9IjUxpFB4Q==";
};

// 然后将你原本的 zone 加入底下这一段宣示
        zone "centos.vbird" IN {
                type master;
                file "named.centos.vbird";
                allow-transfer { 192.168.100.10; };
                update-policy {
                        grant web name web.centos.vbird. A;
                };
        };

[root@www ~]# chmod g+w /var/named
[root@www ~]# chown named /var/named/named.centos.vbird
[root@www ~]# /etc/init.d/named restart
[root@www ~]# setsebool -P named_write_master_zones=1
```

注意到上头的 grant web name web.centos.vbird. A; 那一行， grant 后面接的就是 key 的名称，也就是说，我这把 web 的 key 在这个 zone (centos.vbird) 里面可以修改主机名 web.centos.vbird 的 A 的标志，亦即是修改主机的 IP 对应啦！语法也就是： grant [key_name] name [hostname] 标签 也就是说，我的一把 key 其实可以给予多种权限喔！就看你如何规范了。

设定好之后，由于未来客户端传来的信息是由我们主机的 named 所写入， 写入的目录在 /var/named/ 当中，所以你必需要修改一下权限喔！ 给他重新启动 DNS，然后观察一下 /var/log/messages 里面有没有错误即可！ 如此一来，DDNS 主机端就设定妥当啰！

- 2. Client 端的更新：

接下来则是 DDNS Client 端的更新了。首先，你必须要由 Server 端取得刚刚建立的那两个档案，请将刚刚建立的 Kweb.+157+36124.key 及 Kweb.+157+36124.private 利用 SSH 的 sftp 传送到客户端，亦即是那部 web.centos.vbird 主机上头，假设你已经将这两个档案放置到 /usr/local/ddns 里面去，然后测试看看：

```
[root@web ~]# cd /usr/local/ddns
[root@web ddns]# nsupdate -k Kweb.+157+36124.key
&gt; server 192.168.100.254
&gt; update delete web.centos.vbird                    &lt;==删除原有的
&gt; update add web.centos.vbird 600 A 192.168.100.200 &lt;==更新到最新的
&gt; send
&gt; 最后在此按下 [ctrl]+D 即可
```

请注意到『 update add web.centos.vbird 600 A 192.168.100.200 』这行，他的意义说的是，新增一笔数据，ttl 是 600 ，给予 A 的标签，对应到 192.168.100.200 的意思～ 至于 nsupdate -k 后面加的则是我们在 Server 端产生的那个 key 档案！

然后你就会发现到在 DNS 服务器端的 /var/named/ 里面多出一个暂存档，那就是 named.centos.vbird.jnl 当然，/var/named/named.centos.vbird 就会随着客户端的要求而更新数据喔！

由于手动更新好像挺麻烦的，我们就让 Client 自动更新吧！利用底下这个 script 即可！

```
[root@web ~]# vim /usr/local/ddns/ddns_update.sh
#!/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
export PATH

# 0\. keyin your parameters
basedir="/usr/local/ddns"                   # 基本工作目录
keyfile="$basedir"/"Kweb.+157+36124.key"    # 将档名填进去吧！
ttl=600                                     # 你可以指定 ttl 的时间喔！
outif="eth0"                                # 对外的联机接口！
hostname="web.centos.vbird"                 # 你向 ISP 取得的那个主机名啦！
servername="192.168.100.254"                # 就是你的 ISP 啊！

# Get your new IP
newip=`ifconfig "$outif" &#124; grep 'inet addr' &#124; \
        awk '{print $2}' &#124; sed -e "s/addr\://"`
checkip=`echo $newip &#124; grep "^[0-9]"`
if [ "$checkip" == "" ]; then
        echo "$0: The interface can't connect internet...."
        exit 1
fi

# create the temporal file
tmpfile=$basedir/tmp.txt
cd $basedir
echo "server $servername"                        &gt;  $tmpfile
echo "update delete $hostname A "                &gt;&gt; $tmpfile
echo "update add    $hostname $ttl A $newip"     &gt;&gt; $tmpfile
echo "send"                                      &gt;&gt; $tmpfile

# send your IP to server
nsupdate -k $keyfile -v $tmpfile
```

你只要将上述的程序里面，特殊字体的部分给他修改一下，就能够以 /etc/crontab 的方式在你的系统内自动执行了！这支程序你也可以在底下的连结下载：

- http://linux.vbird.org/linux_server/0350dns/ddns_update.sh

利用 BIND 9 所提供的这个服务，我们只要具有一组固定的 IP ，并向 ISP 申请一个合法授权的 domain name， 就可以提供不论是固定或者是非固定的 IP 使用者，一个合法的主机名了！ 并且，使用者也可以自行透过 nsupdate 来修改自己的 IP 对应！以让自己的主机 IP 永远与主机名保持正确的对应！这对只有拨接制上网的用户来说，真是方便啊！

19.7 重点回顾

- 在 Internet 当中，任何一部合法的主机都具有独一无二的主机名，这个主机名包含了 hostname 与 domain name ，并称为 Fully Qualified Domain Name (FQDN)；
- 为了克服人类对于 IP 不易记忆的困扰，而有名称解析器的产生，首先是 /etc/hosts ，而后则是 DNS 系统的产生；
- 目前 Unix Like 的机器当中，都是以 BIND 这个柏克莱大学发展的软件来架设 DNS 服务器；
- DNS 是个协议的名称，BIND 则是一个软件，这个软件提供的程序为 named ！
- 在 DNS 当中，每一笔记录我们就称他为 RR (Resource Record)。
- 在 DNS 系统中，正解为由 hostname 找 IP ，而反解则是由 IP 找 hostname ，至于 zone 则是一个或者是部分网域的设定值；
- 在 bind 9 之后，预设的情况下 named 已经作了 chroot 的动作。
- Slave 主机本身并没有自行设定 zone file ，其 zone file 是由 Master 主机传送而来，因此， master 主机必须要针对 slave 主机开放 allow-transfer 的设定项目才行。
- 整个 DNS 搜寻的流程当中，若找不到本身的数据，则会向 root(.) 要求资料；
- 正解的纪录(record)主要有：SOA, A, MX, NS, CNAME, TXT 及 HINFO 等；
- 反解的纪录主要有： SOA, PTR 等；
- DNS 查询的指令主要有： host, nslookup, dig, whois 等等；
- 在载入了 named 这个 daemon 之后，请务必前往 /var/log/messages 察看此 daemon 的成功与否。

19.8 本章习题

- 为何要有 DNS 系统：最主要的功能其实在于 Hostname 对应 IP 的查询，可以让我们人类以计算机主机名连上 Internet ，而不必背诵 IP 哩！
- Unix Like 系统当中，主要使用那个软件做为 DNS 主机的架设，同时，他又是使用那个 daemon 来启动 DNS 系统？在 Unix Like 系统当中，使用 BIND 这个软件做为 DNS 的架设，至于 daemon 则是使用 named 这个 daemon ！
- 最早的 Internet 其实是为了政府人员可以连上网络以进行资源的分享，另外，则是电子邮件的使用。 而在早期使用的重要档案只有 /etc/hosts 这个，请教这个 hosts 档案的内容含有什么项目？这个档案的『格式』为『 [IP] [主机名] [主机别名(aliase)]』，而，这个档案里面放置了至少一行，也就是： 127.0.0.1 localhost localhost.localdomain 另外，也可以将经常连接的主机 IP 与 HOSTNAME 的对应给他写进来！
- 正解档案(forward)反解档案(reverse)与内部循环使用的档案(loopback)主要的纪录功能为：正解档在设定 hostname 对应到 IP 的纪录，主要的纪录有 A, NS, SOA, MX, CNAME 等等； 反解档主要设定 IP 对应到 Hostname 的纪录，主要的纪录为 SOA, NS 与 PTR 等。 内部循环则是 localhost 与 127.0.0.1 的对应啦！
- 在主要的 DNS 配置文件 /etc/named.conf 当中，有一个较为特殊的档案，他的类型为 hint ，请问这个档案的功能为何？这个档案主要是由 rs.internic.net 所下载下来的，主要记录了 root (.) 这个 zone 的 IP ！可以让我们的 DNS Server 在找不到数据库时，可以到这个 root 去查询数据！
- 在 client 端搜寻 HOSTNAME 对应到 IP 的查询时，最重要的档案，以及该档案的主要用途为何？/etc/nsswitch.conf ：可以用来设定查询主机名的顺序！例如先查询 /etc/hosts 再查询 DNS 系统； /etc/hosts ：最早的名称解析器； /etc/resolv.conf：这就是 DNS 系统的 resolver (解析器)了。
- 一般来说，在 Client 端使用的查询 HOSTNAME 的指令大多使用什么？nslookup ：可以用来收集一部主机的相关信息； dig：可以用来收集详细的主机信息； whois ：可以用来收集详尽的 DNS 主机信息。 host 则较为简单喔！
- 请问 named 重要的信息登录在在那个档案中？在 /var/log/messages 当中

19.9 参考数据与延伸阅读

- 注1：可以找到的最顶层领域名 (gTLD, ccTLD) 相关查询网站：http://www.whois365.com/tw/listtld http://icannwiki.org/GTLD_and_ccTLD
- BIND 官方网站：http://www.isc.org/products/BIND/
- Study Area 学习网站：http://www.study-area.org/linux/servers/linux_dns.htm
- 优客笔记：http://turtle.ee.ncku.edu.tw/~tung/dns/dnsintro.html
- lame server 的简易说明：http://linux.cvf.net/lame_server.html
- DDNS 架设：http://www.study-area.org/tips/ddns.htm
- Hinet 反解申请单：http://hidomain.hinet.net/top1.html
- 合法 DNS 服务器设定的检查网站：http://www.dnsreport.com/
- 对于想要架设内外部不同 DNS 查询功能的朋友来说，可以参考 view 这个参数，请参考： http://www.study-area.org/tips/bind9_view.htm
- 来自 Red Hat 公司的一份教学：http://www.redhat.com/magazine/026dec06/features/dns/?sc_cid=bcm_edmsept_007
- 台湾 NIC 制作的很棒的教学：http://dns-learning.twnic.net.tw/bind/toc.html
- bind 的 view 应用：http://www.l-penguin.idv.tw/article/dns.htm
- 管理 IP 的单位http://rms.twnic.net.tw/twnic/User/Member/Search/main7.jsp?Order=inet_aton%28Startip%29

2002/12/10：首次完成 2003/03/10：修改部分内容，并且新增 LPI 相关性与重点整理部分！ 2003/09/10：修改了部分的版面，并将 slave DNS 的错误修订完毕！ 2003/10/08：新增了 lame server 的说明，与解决之道！ 2004/10/29：新增了 rndckey 的说明与解决之道！ 2004/10/30：新增了 Master/Slave 的架构设定 2004/10/31：新增了 动态 DNS 主机的设定 。 2005/07/19：增加了 SOA 内五个数字的大小 2006/10/17：将之前的旧文章移动到此处 2006/10/20：终于～不容易～将一些数据给他修订完毕啦！ 2007/06/25：小州大大来信告知 Forwarding 与 cache-only 的介绍可以加以修改。已经处理成为这样。 2011/04/26：将旧的基于 CentOS 4.x 的版本移动到此处 2011/05/10：增加了不少东西，包括将 view 也加进来～欢迎大家参考！ 2011/08/04：将基于 CentOS 5.x 的版本移动到此处 2011/08/05：光是要将这一堆东西测试完毕就花好多时间！这一版的 script 有点不一样，得要注意喔！

第二十章、WWW 伺服器

最近更新日期：2011/08/05

我們最常講的『架站』其實就是架設一個 Web 網站啦！那麼什麼是 Web 呢？那就是全球資訊廣播的意思 (World Wide Web)，或者也可以稱之為互連網吧！這個是我們目前的人類最常使用的 Internet 的協定之一啦！通常說的上網就是使用 WWW 來查詢使用者所需要的資訊囉！目前在 Unix-Like 系統中的 WWW 伺服器主要就是透過 Apache 這個伺服器軟體來達成的， 而為了動態網站，於是 LAMP (Linux + Apache + MySQL + PHP) 就這麼產生啦！讓我們趕緊來進入 LAMP 的世界吧！

20.1 WWW 的簡史、資源以及伺服器軟體

你知道網路爲什麼會這麼流行嗎？其實都是 WWW 造成的啦。早在 1993 年左右，鳥哥初次接觸到網路， 當時的網路較熱門的大概就是一些資源下載的 FTP 網站以及很多文字熱烈討論的 BBS 站了。 資料雖然豐富，不過，總是覺得少了點什麼。後來上了研究所，爲了課業需要， 經常連上台灣的學術網路 (TANET) 進行一些學術資料的檢索，當時大約是 1996 年左右。 因爲上網就是要找資料而已，所以就慢慢的很少使用網路了。

過了幾年後，再次使用圖形介面的作業系統，竟然發現只要點幾個小按鈕，就會有很多網路上花花綠綠的文字與圖案， 有的網站甚至提供影音的特效，當時眞是相當的訝異！不過，由於圖形影像的視覺方面要比 BBS 純文字的資料吸引人，自然造成很多人喜歡流連在網際網路上，人潮多當然就有商機！由於奇貨可居，才有後來 90 年代末期的瀏覽器大戰，這個商業大戰也造成後來 WWW 標準不被某些瀏覽器所支援的後果。

這些年由於搜尋引擎、個人網誌 (blog)、社群網站 (例如 facebook 等)、智慧手機等的流行，又將網際網路推向另一個新境界！啊！ 要學的東西眞是很多啊～@_@。底下讓我們來瞭解瞭解什麼是 WWW 以及他所需要的伺服器軟體，還有一些瀏覽器相關的資訊吧！

20.1.1 WWW 的簡史、HTML 與標準制訂 (W3C)

網際網路 (TCP/IP) 會這麼熱門，主要是 80 年代的 email 以及 90 年代之後的 WWW 服務所造成的！尤其是 WWW 這個玩意兒。 WWW 是 World Wide Web 的縮寫，其中 Web 有廣播網的意思存在， 所以簡稱爲全球資訊網的就是了。WWW 可以結合文字、圖形、影像以及聲音等多媒體，並透過可以讓滑鼠點擊的超連結 (Hyperlink) 的方式將資訊以 Internet 傳遞到世界各處去。

與其他的伺服器類似的，你要連結上 WWW 網站時，該網站必需要提供一些資料， 而你的用户端則必需要使用可以解析這些資料的軟體來處理，那就是瀏覽器啦！簡單的來説，你可以這樣瞧一瞧 WWW server/client 的相關性：

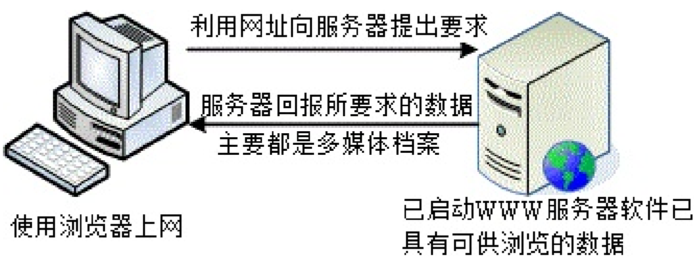


圖 20.1-1、WWW 伺服器與用户端瀏覽器之間的連線相關性

從上面的圖示當中，我們大概可以得到一些觀念：

- WWW 伺服器不但需要一個可讓用户端瀏覽的平台，還需要提供用户端一些資料才行！

- 伺服器所提供的最主要資料是超文件標籤語言 (Hyper Text Markup Language, HTML)、多媒體檔案 (圖片、影像、聲音、文字等，都屬於多媒體或稱爲超媒體)。
- HTML 只是一些純文字資料，透過所謂的標籤 (<tag>) 來規範所要顯示的資料格式；
- 在用户端，透過瀏覽器的對 HTML 以及多媒體的解析，最後呈現在使用者的螢幕上。

- HTML 的格式

如上所提到的相關資訊，我們知道伺服器端需要提供用户端一些資料，而這些資料其實主要都以 HTML 的格式來呈現的。那麼什麼是 HTML 呢？我們拿鳥哥的網站來看一下好了。你可以使用任何一個瀏覽器連結到 http://linux.vbird.org，然後在其上的頁面上按下滑鼠右鍵， 選擇察看原始碼，你就能發現該網頁是如何寫成的了。

```
&lt;!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"&gt;
&lt;html xmlns="http://www.w3.org/1999/xhtml" xml:lang="zh-TW" lang="zh-TW"&gt;
&lt;head&gt;
....一些此頁面的資訊解釋的標頭資料，例如 title 與整體化設計等等....
&lt;/head&gt;
&lt;body style="margin:0; padding:0"&gt;
....在瀏覽器顯示的畫面中，實際放置在瀏覽器上面的資料則寫於此....
&lt;/body&gt;
&lt;/html&gt;
```

HTML 之所以被稱爲標籤語言就如同上面的表格所示，他是由很多 <tag> 所組成的，除了 <!DOCTYPE> 的部分是在宣告底下的語法應該用第幾版的 HTML 解析之外，HTML 主要是由 <html> </html> 所包含起來， 而在其中又分爲兩大區塊，一個是與標頭有關的 <head> </head> 區塊， 包括該網頁所使用的編碼格式與抬頭等等。另一部份則是 <body> </body> 所含有的實際網頁內容資料啦。

HTML 不在本文的介紹內，你可以在市面上找到很多相關的書籍。而傳統的 HTML 4 實際上已經不足以滿足某些美工人員及程式設計師的需求，因此，目前還有改善 HTML 顯示的 CSS 樣式表， 可以讓很多程式互相取用的 XML，還有最新一代的 HTML5 等等，都值得參考喔。

- WWW 所用的協定及 WWW 伺服器簡史--就是講古時間

知道了 WWW 的 server/client 架構後，再來我們要討論的是，那 WWW 是怎麼來的啊？ 伯納斯-李 (Tim Berners-Lee) 在 1980 年代爲了更有效率的讓歐洲核子物理實驗室的科學家可以分享及更新他們的研究成果， 於是他發展出一個超文件傳輸協定 (Hyper Text Transport Protocol, HTTP)。 如同前面提到的，在這個協定上面的伺服器需要軟體，而用户端則需要瀏覽器來解析伺服器所提供的資料。 那麼這些軟體怎麼來的？

爲了讓 HTTP 這個協定得以順利的應用，大約在 90 年代初期由伊利諾大學的國家超級電腦應用中心 (NCSA, http://www.ncsa.illinois.edu/) 開發出伺服器 HTTPd (HTTP daemon 之意)。 HTTPd 爲自由軟體，所以很快的領導了 WWW 伺服器市場。 後來網景通訊 (Netscape) 開發出更強大的伺服器與相對應的用户端瀏覽器，那就是大家曾經熟悉的 Netscape 這套軟體啦。這套軟體分爲伺服器與瀏覽器，其中瀏覽器相對便宜，不過伺服器可就貴的嚇人了。 所以，在伺服器市場上主要還是以 HTTPd 爲主的。

後來由於 HTTPd 這個伺服器一直沒有妥善的發展，於是一群社群朋友便發起一個計畫，這個計畫主要在改善原本的 HTTPd 伺服器軟體，他們稱這個改良過的軟體爲 Apache，取其『一個修修改改的伺服器 (A patch server)』的雙關語！ ^_^！這個 Apache 在 1996 年以後便成爲 WWW 伺服器上市佔率最高的軟體了 (http://httpd.apache.org/)。

- 瀏覽器 (browser) 大戰與支援的標準

雖然 WWW 越來越重要，但相對的來說，用戶端如果沒有瀏覽器的話那麼他們當然就無法去瀏覽 WWW 伺服器所提供的資料。爲了搶佔瀏覽器的市佔率，於是在 90 年代末期微軟將 IE 瀏覽器內建在 Windows 作業系統內，此一決定也讓當時相當廣泛使用的 Netscape 瀏覽器 (Navigator) 市佔率急速下降。 後來網景公司在 1998 年左右將瀏覽器的原始碼部分開放成爲自由軟體，採用 Mozilla 通用授權 (MPL) 。

Mozilla (http://www.mozilla.org/) 這個計畫所開發的軟體可不止瀏覽器而已，還包括郵件處理軟體及網頁編輯軟體等等。 當然啦，其中最出名的就是瀏覽器軟體『火狐狸 (firefox)』啦！那這玩意兒與 IE 有啥不同？ 由於 IE 是整合在 Windows 作業系統核心內， 加上改版的幅度太慢，甚至 IE 使用的 HTML 標準語法解析行爲都是微軟自訂的標準， 並不全然符合網際網路上的標準規範 (W3C, http://www.w3.org/)，導致伺服器端所提供的資料並無法在所有的瀏覽器上都顯示出相同的樣式， 而且用户端也容易受到網路攻擊。

firefox (http://moztw.org/) 的發展就標榜小而美，因此程式相當的小，所以執行效能上面非常的快速，此外，對於超文件的解析上面， firefox 主要依據 w3c 所制訂的標準來發展的，所以任何以 w3c 的標準開發的網站，在 firefox 上面就能夠得到設計者所希望的樣式！目前 firefox 已經針對市面上最常見到的 Windows/Linux/Unix 等作業系統來進行支援，大家可以多多使用喔！^_^

而爲了加快 javascript 的程式運作，並且加快瀏覽的速度，Google 自己也推出一個瀏覽器，稱爲 chrome 瀏覽器， 這個瀏覽器就如 google 的搜尋引擎一般，強調的就是快速！快速！更快速！因此，如果你想要瀏覽器不要花花綠綠， 就是風格簡約，強調速度感，那麼 google 的這個 chrome 自由軟體瀏覽器也可以玩玩的！

由上面的介紹我們可以稍微歸納一下：

- WWW 是依據 HTTP 這個協定而來的，分爲伺服器端與用户端；
- Apache 是一個伺服器端的軟體，主要依據 NCSA 的 HTTPd 伺服器發展而來，爲自由軟體；
- Mozilla 是一個自由軟體的開發計畫，其中 firefox 瀏覽器是相當成功的作品。
- 在撰寫自己的網頁資料時，盡量使用 W3C 所發佈的標準，這樣在所有的瀏覽器上面才能夠順利的顯示出你想要的樣子。

20.1.2 WWW 伺服器與瀏覽器所提供的資源設定 (URL)

現在我們知道 WWW 伺服器的重點是提供一些資料，這些資料必需要是用户端的瀏覽器可以支援顯示才行。 那麼這些資料是什麼類型啊？很簡單啊，當然大部分就是檔案囉。如此說來，我們必需要在伺服器端先將資料檔案寫好， 並且放置在某個特殊的目錄底下，這個目錄就是我們整個網站的首頁了！ 一般來說，這個目錄很可能是在 /var/www/html/ 或者是 /srv/www/。我們的 CentOS 預設在 /var/www/html 喔。

那麼瀏覽器如何取得這個目錄内的資料呢？你必需要在瀏覽器的『網址列』輸入所需要的網址才行。 這個網址就對應到 WWW 伺服器的某個檔案檔名就是了。不過，現今的瀏覽器功能實在很多，他不只可以連上 WWW ，還可以連上類似 FTP 之類的網路通訊協定。所以你得要在網址列輸入正確的網址，這個網址包括這樣：

- <協定>://<主機位址或主機名稱>[:port]/<目錄資源>
- 網址列的意義

上頭就是我們常常聽到的 URL (Uniform Resource Locator) 啦！以斜線作爲分段， 它可以這樣被解釋：

- 協定：

  瀏覽器比較常支援的協定有 http, https, ftp, telnet 等等，還有類似 news, gopher 等。 這個協定在告知瀏覽器『請你利用此一協定連接到伺服器端』的意思。舉例來說，如果你下達： http://ftp.ksu.edu.tw 這表示瀏覽器要連結到 崑山科大的 http (亦即 port 80) 的意思。如果是 ftp://ftp.ksu.edu.tw 則代表連結到 ftp (port 21) 啦！ 因爲使用的協定不同，所以當然回應的資料也不相同的。不過， 萬一對方伺服器的埠口啓動在非正規的埠號，例如將 http 啓動在 port 81 時，那你就得要這樣寫： http://hostname:81/ 。

- 主機位址或主機名稱：

  就是伺服器在網際網路所在的 IP 位置。如果是主機名稱的話，當然得要透過名稱解析器囉！ 一般來說，雖然使用 IP 就能夠架設 WWW 網站，不過建議你還是申請一個好記又合法的主機名稱比較好！

- 目錄資源：：

  剛剛不是提到首頁的目錄嗎？在首頁目錄下的相對位置就是這個目錄資源啦。 舉例來說，鳥哥的網站 www 資料放置在我主機的 /var/www/html/ 當中，所以說：

  - http://linux.vbird.org --> /var/www/html/
  - http://linux.vbird.org/linux_basic/index.php --> /var/www/html/linux_basic/index.php
    另外，通常首頁目錄底下會有個特殊的檔案名稱，例如 index.html 或 index.??? 等。舉例來說，如果你直接按下： http://linux.vbird.org 會發現其實與 http://linux.vbird.org/index.php 是一樣的！ 這是因爲 WWW 伺服器會主動的以該目錄下的『首頁』來顯示啦！

所以啦，我們的伺服器會由於瀏覽器傳來的要求協定不同而給予不一樣的回應資料。那你瞭解到網址列的意義了嗎？

- WWW server/client 間資料傳輸的方式

如果瀏覽器是以 http://hostname 的型態來向伺服器要資料時，那麼瀏覽器與伺服器端是如何傳遞資料的呢？ 基本上有這幾種方法：

- GET 就是瀏覽器直接向 WWW 伺服器要求網址列上面的資源，這也是最常見的。此外，使用 GET 的方式可以直接在網址列輸入變數喔。舉例來說，鳥哥的討論區有一篇提問的智慧， 他的網址是：『http://phorum.vbird.org/viewtopic.php?t=96』，發現那個 ?t=96 了嗎？ t 就是變數， 96 就是這個變數的內容。如果你將問號後面的資料拿掉時，瞧瞧會出現什麼後果？ 這麼說，你可以明白 GET 的處理了吧？
- POST 這也是用户端向伺服器端提出的要求，只是這個要求裡面含有比較多的資料就是了。 舉例來說，討論區裡面不是常常有留言的選項嗎，如果你選擇留言的話不是會在瀏覽器冒出一個框框讓你填入資料嗎！ 當按下傳送後，那些框框內的資料就會被瀏覽器包起來傳送至 WWW 伺服器了。 POST 與 GET 不相同喔， GET 可以在網址列取得用户端所要求的變數，不過 POST 就不是使用網址列的功能了。
- HEAD 伺服器端回應給 Client 端的一些資料檔頭而已；
- OPTIONS 伺服器端回應給 Client 端的一些允許的功能與方法；
- DELETE 刪除某些資源的舉動。

常見的是 GET 這個項目啦！如果有大量資料由用户端上傳到 WWW 伺服器端時，才會使用到 POST 這個項目。 你還是得需要注意一下這些舉動，因爲後續的登錄檔分析內容都是使用這種動作來分析的呦！

20.1.3 WWW 伺服器的類型： 系統、平台、資料庫與程式 (LAMP)

以目前的網路世界來說，市佔率較高的 WWW 伺服器軟體應該是 Apache 與 IIS 這兩個玩意兒， Apache 是自由軟體，可以在任何作業系統上面安裝的，至於 IIS 則是 Windows 家族開發出來的， 僅能在 Windows 作業系統上面安裝與執行。由於作業系統平台不一樣，所以其上安裝的軟體當然也不相同。 底下就讓我們來聊一聊目前網站的一些特色吧！

- 僅提供使用者瀏覽的單向靜態網頁

這種類型的網站大多是提供『單向靜態』的網頁，或許有提供一些動畫圖示，但基本上就僅止於此啦！ 因爲單純是由伺服器單向提供資料給客户端，Server 不需要與 Client 端有互動，所以你可以到該網站上去瀏覽， 但是無法進行進行資料的上傳喔！目前主要的免費虛擬主機

大多是這種類型。所以，你只要依照 HTML 的語法寫好你的網頁，並且上傳到該網站空間上，那麼你的資料就可以讓大家瀏覽了！

- 提供使用者互動介面的動態網站

這種類型的網站可以讓伺服器與使用者互動，常見的例如討論區論壇與留言版，包括一些部落格也都是屬於這類型。 這類型的網站需要的技術程度比較高，因爲他是藉由『網頁程式語言』來達成與使用者互動的行爲， 常見的例如 PHP 網頁程式語言，配合 MySQL 資料庫系統來進行資料的讀、寫。整個互動可以使用下圖來說明：

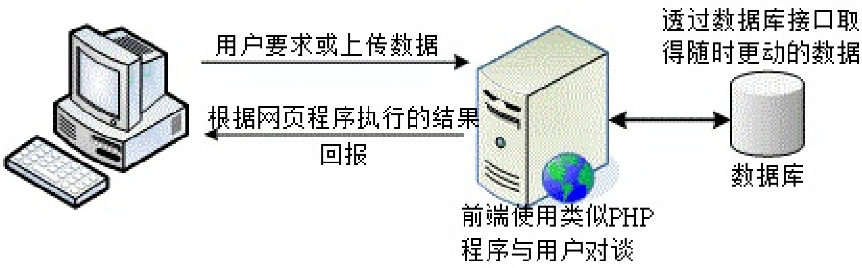


圖 20.1-2、動態網站的網頁程式語言與資料庫介面

這就是所謂的伺服器端工作任務介面 (Server Side Include, SSI)，因爲不論你要求的資料是什麼，其實都是透過伺服器端同一支網頁程式在負責將資料讀出或寫入資料庫， 處理完畢後將結果傳給用户端的一種方式，變動的是資料庫內的資料，網頁程式其實並沒有任何改變的。這部份的網頁程式包括 PHP, ASP, Perl...很多啦！

另外一種互動式的動態網頁主要是在用户端達成的！舉例來說，我們可以透過利用所謂的 Java scripts 這種語法， 將可執行的程式碼 (java script) 傳送給用户端，用户端的瀏覽器如果有提供 java script 的功能， 那麼該程式就可以在用户端的電腦上面運作了。由於程式是在用户端電腦上執行， 因此如果伺服器端所製作的程式是惡意的，那麼用户端的電腦就可能會遭到破壞。 這也是爲啥很多瀏覽器都已經將一些危險的 java script 關閉的原因。

另外一種可在用户端執行的就是 flash 動畫格式，在這種動畫格式內還可以進行程式設計，因此用户端只要擁有可以執行 flash 動畫的軟體，那就可以利用這個軟體來達到互動式的對談。 這些都算是動態網站所提供的功能喔！

從上面的說明你可以知道動態網站是目前比較熱門的，像是近兩年來如同雨後春筍一般冒出來的個人部落格 (blog) 就是很經典的動態網站之一。而由圖 20.1-2 我們也知道要做成這樣的動態網站你必需要有：

- 支援的作業系統：讓所需要的軟體都能夠安裝執行啊；
- 可運作的 WWW 伺服器：例如 Apache 與 IIS 等 WWW 伺服器平台軟體；
- 網頁程式語言：包括 perl, PHP, JSP, CGI, ASP 等等都算是啦！
- 資料儲存之資料庫系統：包括 MySQL, MSSQL, PostgreSQL 以及甲骨文 (Oracle) 等等。

- LAMP 平台的說明

在整個平台設計上面，目前常見的有兩大系統，一個是 Linux 作業系統上面，搭配 Apache + MySQL + PHP 等而達成，這個系統被稱爲 LAMP。另一個則是微軟的 IIS + MSSQL + ASP (.NET) 伺服器。在能見度與市佔率方面，應該還是以 LAMP 爲主吧！在 LAMP 裡面除了 Linux 之外，其他三個小東西就讓我們來談談先：

- Apache (http://www.apache.org)

  1995 年以前就有很多的 WWW 伺服器軟體，其中以 HTTPd 佔有率較高。 後來 HTTPd 經過多次臭蟲的修訂後，才在 1995 年後發佈 Apache (A patch server) 的啦！這東西就是主要提供 WWW 的伺服器平台，後面談到的 PHP 必須要在這玩意兒上才能運作！

- MySQL (http://www.mysql.org/)

  傳統的檔案讀取是很麻煩的，如果你只要讀取該檔案當中的一小部分，系統還是會將整個檔案讀出來， 若又有多人同時讀取同一個檔案時，那就會造成效能與系統上的問題，所以才會有資料庫系統的推出。 資料庫其實是一種特殊格式的檔案，這種檔案必需要透過特殊介面 (資料庫軟體) 來進行讀寫。由於這個特殊介面已經針對資料的查詢、寫入做過最佳化設計， 因此很適合多人同時寫入與查詢的工作。

  針對資料庫的語法有所謂的 SQL 標準語法，任何根據這種資料檢索語法發展出來的資料庫，就稱爲 SQL 資料庫。 比較知名的自由軟體資料庫系統有 MySQL 及 PostgreSQL ，其中 MySQL 的使用率又比較高一些。 MysQL 可以透過網頁程式語言來進行讀寫的工作，因此很適合例如討論區、論壇等的設計， 甚至很多商業網站的重要資料也是透過 MySQL 這個資料庫軟體來存取的呢！

- PHP (http://www.php.net/)

  按照官方的說法來說，PHP 是一個工具，他可以被用來建立動態網頁，PHP 程式碼可以直接在 HTML 網頁當中嵌入， 就像你在編輯 HTML 網頁一樣的簡單。所以說， PHP 是一種『程式語言』，這種程式語言可以直接在網頁當中編寫， 不需要經過編譯即可進行程式的執行。由於具有：自由軟體、跨平台、容易學習及執行效能高等優點， 目前是很熱門的一個設計網頁的咚咚喔！你可以在市面上找到很多相關的書籍來參考的。

Tips: 事實上，如果光學會 Linux 與架站，對你自己的競爭力還是不夠的，可以的話，多學一些 MySQL 的 SQL 語法，以及類似 PHP, JSP 等跨平台的網頁程式語言，對你的未來是很有幫助的喔！



20.1.4 https： 加密的網頁資料 (SSL) 及第三方公正單位

關於 HTTP 這個傳輸協定當中，你必需要知道的是：『這個傳輸協定傳輸資料是以明碼傳送的』， 所以你的任何資料封包只要被監聽竊取的話，那麼該資料就等於是別人的啦！那想一想， 你有過上線刷卡的經驗嗎？上線刷卡只要輸入你信用卡的卡號與相關的截止日期後，就能夠進行交易了。 如果你的資料在 Internet 上面跑時是明碼的情況下，眞要命！那你的信用卡不就隨時可能會被盜用？

雖然大多數 Internet 上面的 WWW 網站所提供的資料是可以隨意瀏覽的，不過如同上面提到的， 一些物流交易網站的資料以及關於你個人的重要機密資料當然就不能這樣隨意傳送啦！ 這個時候就有需要用到 https://hostname 這種連線的方式啦！這種方式是透過 SSL 加密的機制喔！

- Secure Socket Layer (SSL)

還記得我們在第十一章的 SSH 伺服器當中介紹過他連線的機制吧？ 就是利用非對稱的 key pair (Public + Private kye) 來組成金鑰，然後透過公鑰加密後傳輸， 傳輸到目標主機後再以私鑰來解密，如此一來資料在 Internet 上面跑就以加密的方式， 想當然爾，這些資料自然就比較安全啦！SSL 就是利用在 WWW 傳輸上面的加密方式之一啦！

當瀏覽器端與 WWW 伺服器端同時支援 SSL 的傳輸協定時，在連線階段瀏覽器與伺服器就會產生那把重要的金鑰！ 產生金鑰後就能夠利用瀏覽器來傳送與接收加密過的重要資料啦！要達成這樣的機制， 你的 WWW 伺服器必需要啓動 https 這個重要的傳輸協定，而瀏覽器則必需要在網址列輸入 https:// 開頭的網址，那兩者才能夠進行溝通與連線。要注意的是，在某些很舊的瀏覽器上面是不支援 SSL 的， 所以在那些舊的瀏覽器上就無法達成 https 的連線啦！

- Certificate Authorities (CA)

想一想 SSL 這個機制有什麼問題？他的問題就是：『那把 Public key 是伺服器產生且任何人都能取得的』！這是什麼問題？因爲 public key 可讓任何人取得， 若被釣魚網站取得並且製作一個很類似你網路銀行的網站，並且騙你輸入帳密，要命了！因爲你不知道該網站是詐騙集團製作的， 以爲 https 就是安全的，如此一來，即使你的資料有加密，但結果，在釣魚網站伺服器端還是能夠取得你輸入的帳密啊！ 這個時候就需要第三方公正單位來幫忙啦！

所謂的 CA 就是一個公認的公正單位，你可以自行產生一把金鑰且製作出必要的憑證資料並向 CA 單位註冊 (講到註冊你就要知道...這東西是要錢的意思！)，那麼當用户端的瀏覽器在瀏覽時，該瀏覽器會主動的向 CA 單位確認該憑證是否爲合法註冊過的，如果是的話，那麼該次連線才會建立，如果不是呢？那麼瀏覽器就會發出警告訊息， 告知使用者應避免建立連線啊。 所以說，如此一來 WWW 伺服器不但有公正單位的背書，使用者在建立連線時也比較有保障！

更多關於 SSL 以及 CA 的介紹，可以約略參考一下：

- Apache 的 SSL： http://www.modssl.org/
- CA 組織之一：https://digitalid.verisign.com/server/apacheNotice.htm

20.1.5 用户端常見的瀏覽器

咱們前面談到 WWW 伺服器是 Server/Client 的架構，而用户端使用的軟體就是瀏覽器啊！目前比較知名的自由軟體瀏覽器主要有兩款，包括 Mozilla 基金會管理的 firefox (火狐狸) 以及 Google 自行推出的 chrome。至於市佔率較高的還有 windows 的 IE。

由於瀏覽器可以連結到網際網路上，所以瀏覽器也有可能被攻擊！ 其中由於 IE 直接內嵌至 Windows 的核心當中，所以如果 IE 有漏洞時，對於系統的損害是很大的！ 因此無論如何，請記得『務必要隨時更新到最新版本的瀏覽器』才行。建議你可以使用 firefox 或 chrome 這些小巧玲瓏的瀏覽器啊！

除了視窗介面的瀏覽器軟體之外，其實還有幾個可以在文字介面底下進行瀏覽與網頁下載的程式，分別是：

- links 與 lynx：文字介面的瀏覽器；
- wget：文字介面下使用來擷取檔案的指令。

這幾個指令我們已經在第五章談過了，請自行前往參考喔！

20.2 WWW (LAMP) 伺服器基本設定

從前面的說明當中，我們知道在 Linux 上面要達成網頁伺服器需要 Apache 這套伺服器軟體吶！不過 Apache 僅能提供最基本的靜態網站資料而已，想要達成動態網站的話，那麼最好還是需要 PHP 與 MySQL 的支援才好。所以底下我們將會以 LAMP 作爲安裝與設定的介紹，加油吧！ ^_^

20.2.1 LAMP 所需軟體與其結構

既然我們已經是 Linux 作業系統，而且使用的是號稱完全相容於 Red Hat Enterprise Linux 的 CentOS 版本， 那當然只要利用 CentOS 本身提供的 Apache, PHP, MySQL 即可！不建議你自行利用 tarball 安裝你的 LAMP 伺服器。因爲自行安裝不但手續麻煩，而且也不見得比系統預設的軟體穩定。 除非你有特殊的需求 (例如你的某些 Apache 外掛程式需要較高的版本，或者是 PHP, MySQL 有特殊版本的需求)， 否則請使用 yum 來進行軟體的安裝即可。

那麼我們的 LAMP 需要哪些東西呢？你必需要知道的是，PHP 是掛在 Apache 底下執行的一個模組， 而我們要用網頁的 PHP 程式控制 MySQL 時，你的 PHP 就得要支援 MySQL 的模組才行！所以你至少需要底下幾個軟體：

- httpd (提供 Apache 主程式)
- mysql (MySQL 客户端程式)
- mysql-server (MySQL 伺服器程式)
- php (PHP 主程式含給 apache 使用的模組)
- php-devel (PHP 的發展工具，這個與 PHP 外掛的加速軟體有關)
- php-mysql (提供給 PHP 程式讀取 MySQL 資料庫的模組)

要注意， Apache 目前有幾種主要版本，包括 2.0.x, 2.2.x 以及 2.3.x 等等，至於 CentOS 6.x 則是提供 Apache 2.2.x 這個版本啦。如果你沒有安裝的話，請直接使用 yum 或者是原本光碟來安裝先：

```
# 安裝必要的 LAMP 軟體： php-devel 可以先忽略～
[root@www ~]# yum install httpd mysql mysql-server php php-mysql
```

先來瞭解一下 Apache 2.2.x 這個版本的相關結構，這樣才能夠知道如何處理我們的網頁資料啊！

- /etc/httpd/conf/httpd.conf (主要設定檔) httpd 最主要的設定檔，其實整個 Apache 也不過就是這個設定檔啦！裡面眞是包山包海啊！不過很多其他的 distribution 都將這個檔案拆成數個小檔案分別管理不同的參數。但是主要設定檔還是以這個檔名爲主的！ 你只要找到這個檔名就知道如何設定啦！

- /etc/httpd/conf.d/*.conf (很多的額外參數檔，副檔名是 .conf) 如果你不想要修改原始設定檔 httpd.conf 的話，那麼可以將你自己的額外參數檔獨立出來， 例如你想要有自己的額外設定值，可以將他寫入 /etc/httpd/conf.d/vbird.conf (注意，副檔名一定是 .conf 才行) 而啟動 Apache 時，這個檔案就會被讀入主要設定檔當中了！這有什麼好處？好處就是當你系統升級的時候， 你幾乎不需要更動原本的設定檔，只要將你自己的額外參數檔複製到正確的地點即可！維護更方便啦！
- /usr/lib64/httpd/modules/, /etc/httpd/modules/ Apache 支援很多的外掛模組，例如 php 以及 ssl 都是 apache 外掛的一種喔！ 所有你想要使用的模組檔案預設是放置在這個目錄當中的！
- /var/www/html/ 這就是我們 CentOS 預設的 apache 『首頁』所在目錄啦！當你輸入『http://localhost』時所顯示的資料， 就是放在這個目錄當中的首頁檔 (預設爲 index.html)。
- /var/www/error/ 如果因爲伺服器設定錯誤，或者是瀏覽器端要求的資料錯誤時，在瀏覽器上出現的錯誤訊息就以這個目錄的預設訊息爲主！
- /var/www/icons/ 這個目錄提供 Apache 預設給予的一些小圖示，你可以隨意使用啊！當你輸入『http://localhost/icons/』 時所顯示的資料所在。
- /var/www/cgi-bin/ 預設給一些可執行的 CGI (網頁程式) 程式放置的目錄；當你輸入『http://localhost/cgi-bin/』 時所顯示的資料所在。
- /var/log/httpd/ 預設的 Apache 登錄檔都放在這裡，對於流量比較大的網站來說，這個目錄要很小心， 因爲以鳥哥網站的流量來說，一個星期的登錄檔資料可以大到 700MBytes 至 1GBytes 左右，所以你務必要修改一下你的 logrotate 讓登錄檔被壓縮，否則...
- /usr/sbin/apachectl 這個就是 Apache 的主要執行檔，這個執行檔其實是 shell script 而已， 他可以主動的偵測系統上面的一些設定值，好讓你啓動 Apache 時更簡單！
- /usr/sbin/httpd 呵呵！這個才是主要的 Apache 二進位執行檔啦！
- /usr/bin/htpasswd (Apache 密碼保護) 在某些網頁當你想要登入時你需要輸入帳號與密碼對吧！那 Apache 本身就提供一個最基本的密碼保護方式， 該密碼的產生就是透過這個指令來達成的！相關的設定方式我們會在 WWW 進階設定當中說明的。

至於 MySQL 方面，你需要知道的幾個重要目錄與檔案有：

- /etc/my.cnf 這個是 MySQL 的設定檔，包括你想要進行 MySQL 資料庫的最佳化，或者是針對 MySQL 進行一些額外的參數指定， 都可以在這個檔案裡面達成的！
- /var/lib/mysql/ 這個目錄則是 MySQL 資料庫檔案放置的所在處啦！當你有啓動任何 MySQL 的服務時， 請務必記得在備份時，這個目錄也要完整的備份下來才行啊！

另外，在 PHP 方面呢，你應該也要知道幾個檔案喔：

- /etc/httpd/conf.d/php.conf 那你要不要手動將該模組寫入 httpd.conf 當中？不需要的，因爲系統主動將 PHP 設定參數寫入這個檔案中了！ 而這個檔案會在 Apache 重新啓動時被讀入，所以 OK 的啦！
- /etc/php.ini 就是 PHP 的主要設定檔，包括你的 PHP 能不能允許使用者上傳檔案？能不能允許某些低安全性的標誌等等， 都在這個設定檔當中設定的啦！
- /usr/lib64/httpd/modules/libphp5.so PHP 這個軟體提供給 Apache 使用的模組！這也是我們能否在 Apache 網頁上面設計 PHP 程式語言的最重要的咚咚！ 務必要存在才行！
- /etc/php.d/mysql.ini, /usr/lib64/php/modules/mysql.so 你的 PHP 是否可以支援 MySQL 介面呢？就看這兩個東西啦！這兩個咚咚是由 php-mysql 軟體提供的呢！
- /usr/bin/phpize, /usr/include/php/ 如果你未來想要安裝類似 PHP 加速器以讓瀏覽速度加快的話，那麼這個檔案與目錄就得要存在， 否則加速器軟體可無法編譯成功喔！這兩個資料也是 php-devel 軟體所提供的啦！

基本上我們所需要的幾個軟體他的結構就是這樣啦！上面提到的是 Red Hat 系統 (RHEL, CentOS, FC) 所需的資料， 如果是 SuSE 或其他版本的資料，請依照你的 distribution 管理軟體的指令 (rpm 或 dpkg) 去查詢一下， 應該就能夠知道各個重要資料放置在哪裡啦！這些資料很重要，你必需要對放置的地點有點概念才行喔！

20.2.2 Apache 的基本設定

在開始設定 Apache 之前，你要知道由於主機名稱對於 WWW 是有意義的，所以雖然利用 IP 也能架設 WWW 伺服器， 不過建議你還是申請一個合法的主機名稱比較好。 如果是暫時測試用的主機所以沒有主機名稱時，那麼至少確定測試用主機名稱爲 localhost 且在你的 /etc/hosts 內需要有一行：

```
[root@www ~]# vim /etc/hosts
127.0.0.1   localhost.localdomain localhost
```

這樣在啓動你的 Apache 時才不會發生找不到完整主機名稱 (FQDN) 的錯誤訊息。此外，Apache 只是個伺服器平台而已，你還需要瞭解 HTML 以及相關的網頁設計語法， 如此才能豐富你的網站。對於想要設計網頁的朋友來說，應用軟體或許是很好入門， 不過想要完整的瞭解網站設計的技巧，還是研究一下基礎的 HTML 或 CSS 比較妥當。

如果你眞的對於一些基礎語法有興趣，並且也想要開發一些所謂的『無障礙網頁空間』的話，那麼可以造訪一下 http://www.w3c.org 所列舉的標準語法，或者是行政院的無障礙網頁空間申請規範 (http://www.webguide.nat.gov.tw) 相信會有所收穫的啦！

終於要來談一談如何設定 Apache 這個 httpd.conf 設定檔了！再次強調，每個 distribution 的這個檔案內容都不很相同，所以你必需要自行找出相關的設定檔才行喔！那麼這個 httpd.conf 的設定爲何呢？他的基本設定格式是這樣的：

```
&lt;設定項目&gt;
    此設定項目內的相關參數
    ...........
&lt;/設定項目&gt;
```

舉例來說，如果你想要針對我們的首頁 /var/www/html/ 這個目錄提供一些額外的功能，那麼：

```
&lt;Directory "/var/www/html"&gt;
    Options Indexes
    ...........
&lt;/Directory&gt;
```

幾乎都是這樣的設定方式喔！特別留意的是，如果你有額外的設定時，不能隨便在 httpd.conf 裡頭找地方寫入！否則如果剛好寫在 <Directory>...</Directory> 裡面，呼呼！ 那麼就會發生錯誤啦！需要前前後後的找一找喔！或者是在檔案的最後面加入也行！好啦，底下咱們先來聊一聊 Apache 伺服器的基礎設定吧！

Tips: 事實上在 Apache 的網頁有提供很多詳細的文件資料，眞的是很詳細啦！ 鳥哥在底下僅是介紹一些慣用的設定項目的意義而已。有興趣的話，請務必要前往查閱： Apache 2.2 核心文件：http://httpd.apache.org/docs/2.2/mod/core.html



- 針對伺服器環境的設定項目

Apache 針對伺服器環境的設定項目方面，包括回應給用户端的伺服器軟體版本、主機名稱、伺服器設定檔頂層目錄等。 底下咱們就來談一談：

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
ServerTokens OS
# 這個項目在僅告知用户端我們伺服器的版本與作業系統而已，不需要更動他。
# 如果不在乎你系統的資訊被遠端的用户查詢到，則可以將這個項目註解掉即可(不建議)

ServerRoot "/etc/httpd"
# 伺服器設定的最頂層目錄，有點類似 chroot 那種感覺。包括 logs, modules
# 等等的資料都應該要放置到此目錄底下 (&lt;u&gt;若未宣告成絕對路徑時&lt;/u&gt;)

PidFile run/httpd.pid
# 放置 PID 的檔案，可方便 Apache 軟體的管理啦！只有相對路徑吧！
# 考慮 ServerRoot 設定值，所以檔案在 /etc/httpd/run/httpd.pid ！

Timeout 60
# 不論接收或傳送，當持續連線等待超過 60 秒則該次連線就中斷。
# 一般來説，此數值在 300 秒左右即可，不需要修改這個原始值啦。

KeepAlive On    &lt;==最好將預設的 Off 改爲 On 啦！
# 是否允許持續性的連線，亦即一個 TCP 連線可以具有多個檔案資料傳送的要求。
# 舉例來説，如果你的網頁內含很多圖檔，那麼這一次連線就會將所有的資料送完，
# 而不必每個圖檔都需要進行一次 TCP 連線。預設爲 Off 請改爲 On 較佳。

MaxKeepAliveRequests 500  &lt;==可以將原本的 100 改爲 500 或更高
# 與上個設定值 KeepAlive 有關，當 KeepAlive 設定爲 On 時，則這個數值可決定
# 該次連線能夠傳輸的最大傳輸數量。爲了增進效能則可以改大一點！0 代表不限制。

KeepAliveTimeout 15
# 在允許 KeepAlive 的條件下，則該次連線在最後一次傳輸後等待延遲的秒數。
# 當超過上述秒數則該連線將中斷。設定 15 差不多啦！如果設定太高 (等待時間較長)，
# 在較忙碌的系統上面將會有較多的 Apache 程序佔用資源，可能有效能方面的困擾。

&lt;IfModule prefork.c&gt;   &lt;==底下兩個 perfork, worker 與記憶體管理有關！
StartServers       8   &lt;==啓動 httpd 時，喚醒幾個 PID 來處理服務的意思
MinSpareServers    5   &lt;==最小的預備使用的 PID 數量
MaxSpareServers   20   &lt;==最大的預備使用的 PID 數量
ServerLimit      256   &lt;==伺服器的限制
MaxClients       256   &lt;==最多可以容許多少個用户端同時連線到 httpd 的意思！
MaxRequestsPerChild  4000
&lt;/IfModule&gt;
&lt;IfModule worker.c&gt;
StartServers         4
MaxClients         300
MinSpareThreads     25
MaxSpareThreads     75
ThreadsPerChild     25
MaxRequestsPerChild  0
&lt;/IfModule&gt;
```

上面的 prefork 及 worker 其實是兩個與伺服器連線資源有關的設定項目。預設的項目對於一般小型網站來説已經很夠用了， 不過如果你的網站流量比較大時，或許可以修訂一下裡面的數值呢！這兩個模組都是用在提供使用者連線的資源 (process)，設定的數量越大代表系統會啓動比較多的程序來提供 Apache 的服務，反應速度就比較快。 簡單的説，這兩個模組的功能分類爲：

- 針對模組的功能分類來説：

  worker 模組佔用的記憶體較小，對於流量較大的網站來説，是一個比較好的選擇。
  prefork 雖然佔用較大的記憶體，不過速度與 worker 差異不大，並且 prefork 記憶體使用設計較爲優秀，可以在很多無法提供 debug 的平台上面進行自我除錯，所以，預設的模組就是 prefork 這一個呢！

- 細部設定的內容方面：(以 Prefork 爲例， worker 意義相同)
    - StartServers：代表啓動 Apache 時就啓動的 process 數量，所以 apache 會用到不止一支程序！
    - MinSpareServers, MaxSpareServers：代表最大與最小的備用程序數量。
    - MaxClients：最大的同時連線數量，也就是 process 不會超過此一數量。現在假設有 10 個人連上來，加上前面的 MinSpareServer=5, MaxSpareServers=20，則 apache 此時的程序數應有 15-30 個之意。而這個最終程序數不可超過 256 個 (依上述設定值)！
    - MaxRequestsPerChild：每個程序能夠提供的最大傳輸次數要求。 舉例來說，如果有個使用者連上伺服器後(一個 process)，卻要求數百個網頁，當他的要求數量超過此一數值， 則該程序會被丟棄，另外切換一個新程序。這個設定可以有效的控管每個 process 在系統上的『存活時間』。 因爲根據觀察所得，新程序的效能較佳啦！

在上面的設定中，比較有趣的是 MaxClients 這個程序模組的參數值，如同上面的說明，這個 MaxClients 設定值可以控制『同時連上 WWW 伺服器的總連線要求』數量，亦即想成最高即時線上人數啦。 不過你要注意的是，MaxClients 的數量不是越高越好，因爲他會消耗實體記憶體 (與 process 有關嘛)，所以如果你設定太高導致超出實體記憶體能夠容許的範圍，那麼效能反而會降低 (因爲系統會使用速度較慢的 swap 啊)，此外， MaxClients 也在 Apache 編譯時就指定最大值了，所以你也無法超出系統最大值， 除非...你重新編譯 Apache 啦！

除非你的網站流量特別大，否則預設值已經夠你使用的了。而如果你的記憶體不夠大的話，那麼 MaxClients 反而要調小一點，例如 150 ，否則效能不佳。那，apache 到底是使用那個模組啊？prefork 還是 worker ？事實上 CentOS 將這兩個模組分別放到不同的執行檔當中，分別是：

- /usr/sbin/httpd：使用 prefork 模組；
- /usr/sbin/httpd.worker：使用 worker 模組。

那如何決定你使用的是哪一支程式？你可以去查閱一下 /etc/sysconfig/httpd ， 就能夠知道系統預設提供 prefork 模組，但你可以透過修改 /etc/sysconfig/httpd 來使用 worker 模組的。 如果你很有好奇心，那麼可以分別試著啓動這兩種模組啊！接下來，繼續瞧瞧其他的伺服器環境設定參數吧！

```
Listen 80
# 與監聽介面有關，預設開放在所有的網路介面啊！也可修改埠口，如 8080

LoadModule auth_basic_module modules/mod_auth_basic.so
....(底下省略)....
# 載入模組的設定項目。Apache 提供很多有用的模組 (就是外掛) 給我們使用了！

Include conf.d/*.conf
# 因爲這一行，所以放置到 /etc/httpd/conf.d/*.conf 的設定都會被讀入！

User apache
Group apache
# 前面提到的 prework, worker 等模組所啓動的 process 之擁有者與群組設定。
# 這個設定很重要，因爲未來你提供的網頁檔案能不能被瀏覽都與這個身份有關啊！

ServerAdmin vbird@www.centos.vbird  &lt;==改成你自己的 email 吧
# 系統管理員的 email，當網站出現問題時，錯誤訊息會顯示的聯絡信箱(錯誤回報)。

ServerName www.centos.vbird    &lt;==自行設定好自己的主機名稱較佳！
# 設定主機名稱，這個值如果沒有指定的話，預設會以 hostname 的輸出爲依據。
# 千萬記得，你填入的這個主機名稱要找的到 IP 喔！(DNS 或 /etc/hosts)

UseCanonicalName Off
# 是否使用標準主機名稱？如果你的主機有多個主機名稱，若這個設定爲 On，
# 那麼 Apache 只接受上頭 servername 指定的主機名稱連線而已。請使用 Off。
```

在某些特殊的伺服器環境中，有時候你會想要啓動多個不同的 Apache，或者是 port 80 已經被使用掉了，導致 Apache 無法啓動在預設的埠口。那麼你可以透過 Listen 這個設定值來修改埠口喔！這也是個很重要的設定值。 此外，你也可以將自己的額外設定指定到 /etc/httpd/conf.d/*.conf 內，尤其是虛擬主機很常使用這樣的設定，在移機時會很方便的！

- 針對中文 big5 語系編碼的設定參數修改

目前的網際網路傳輸資料編碼多是以萬國碼 (UTF-8) 爲主，不過在台灣還是有相當多的網站使用的是 Big5 的繁體中文編碼啊！如果你的 Apache 預設是以 UTF-8 編碼來傳輸資料，但你 WWW 的資料卻是 big5 ， 那麼用户端將會看到『亂碼』！雖然可以透過調整瀏覽器的編碼來讓資料正確顯示，不過總是覺得很討厭。 此時，你應該可以調整一下底下的參數喔！

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
# 找到底下這一行，應該是在 747 行左右
# AddDefaultCharset UTF-8  &lt;==請將她註解掉！
```

這個設定值的意義是説，讓伺服器傳輸『強制使用 UTF-8 編碼』的訊息給用户端瀏覽器，因此不論網頁內容寫什麼， 反正在用户端瀏覽器都會預設使用萬國馬來顯示的意思。那如果你的網頁使用的是非萬國碼的語系編碼， 此時就會在瀏覽器內出現亂碼了！非常討厭～所以這裡當然需要註解掉。 你必須要注意的是，如果你已經在用户端上面瀏覽過許多頁面，那麼你修改過這個設定值後，仍然要將瀏覽器的快取 (cache) 清除才行！否則相同頁面仍可能會看到亂碼！網友們已經回報過很多次了， 這不是 Apache 的問題，而是用户端瀏覽器的快取所產生的啦！記得處理處理！

語系編碼已經取消預設值，那我怎麼知道我的網頁語系在用户端會顯示的是哪一個？其實在網頁裡面本來就有宣告了：

```
&lt;html&gt;
&lt;head&gt;
    &lt;meta http-equiv="Content-Type" content="text/html; charset=big5" &gt;
    ....(其他省略)....
```

你應該要修訂的是上述的特殊字體處，而不是透過 Apache 提供預設語系才對！

- 網頁首頁及目錄相關之權限設定 (DocumentRoot 與 Directory)

我們不是講過 CentOS 的 WWW 預設首頁放置在 /var/www/html 這個目錄嗎？爲什麼呢？因爲 DocumentRoot 這個設定值的關係啦！此外，由於 Apache 允許 Internet 對我們的資料進行瀏覽， 所以你當然必須要針對可被瀏覽的目錄進行權限的相關設定，那就是 <Directory> 這個設定值的重要特色！先讓我們來看看預設的主網頁設定吧！

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
DocumentRoot "/var/www/html"  &lt;==可以改成你放置首頁的目錄！
# 這個設定值規範了 WWW 伺服器主網頁所放置的『目錄』，雖然設定值內容可以變更，
# 但是必須要特別留意這個設定目錄的權限以及 SELinux 的相關規則與類型(type)！

&lt;Directory /&gt;
    Options FollowSymLinks
    AllowOverride None
&lt;/Directory&gt;
# 這個設定值是針對 WWW 伺服器的『預設環境』而來的，因爲針對『/』的設定嘛！
# 建議保留上述的預設值 (上頭資料已經是很嚴格的限制)，相關參數容後說明。

&lt;Directory "/var/www/html"&gt;         &lt;==針對特定目錄的限制！底下參數很重要！
    Options Indexes FollowSymLinks  &lt;==建議拿掉 Indexes 比較妥當！
    AllowOverride None
    Order allow,deny
    Allow from all
&lt;/Directory&gt;
```

這個地方則是針對 /var/www/html 這個目錄來設定權限啦！就是咱們首頁所在目錄的權限。主要的幾個設定項目的意義是這樣的 (這些設定值都很重要！要仔細看喔！)：

- Options (目錄參數)：

  此設定值表示在這個目錄內能夠讓 Apache 進行的動作，亦即是針對 apache 的程序的權限設定啦！主要的參數值有：

  - Indexes：如果在此目錄下找不到『首頁檔案 (預設爲 index.html) 』時， 就顯示整個目錄下的檔案名稱，至於『首頁檔案檔名』則與 DirectoryIndex 設定值有關。
  - FollowSymLinks：這是 Follow Symbolic Links 的縮寫， 字面意義是讓連結檔可以生效的意思。我們知道首頁目錄在 /var/www/html，既然是 WWW 的根目錄，理論上就像被 chroot 一般！ 一般來說被 chroot 的程式將無法離開其目錄，也就是說預設的情況下，你在 /var/www/html 底下的連結檔只要連結到非此目錄的其他地方，則該連結檔預設是失效的。 但使用此設定即可讓連結檔有效的離開本目錄。

    - ExecCGI：讓此目錄具有執行 CGI 程式的權限，非常重要！舉例來說，之前熱門的 OpenWebMail 使用了很多的 perl 的程式，你要讓 OpenWebMail 可以執行，就得要在該程式所在目錄擁有 ExecCGI 的權限才行喔！但請注意，不要讓所有目錄均可使用 ExecCGI ！
    - Includes：讓一些 Server-Side Include 程式可以運作。建議可以加上去！
    - MultiViews：這玩意兒有點像是多國語言的支援，與語系資料 (LanguagePriority) 有關。最常見在錯誤訊息的回報內容，在同一部主機中，可以依據用户端的語系而給予不同的語言顯示呢！ 預設在錯誤回報訊息當中存在，你可以檢查一下 /var/www/error/ 目錄下的資料喔！

- AllowOverride (允許的覆寫參數功能)：

  表示是否允許額外設定檔 .htaccess 的某些參數覆寫？我們可以在 httpd.conf 內設定好所有的權限，不過如此一來若使用者自己的個人網頁想要修改權限時將會對管理員造成困擾。因此 Apache 預設可以讓使用者以目錄底下的 .htaccess 檔案內覆寫 <Directory> 內的某些功能參數。 這個項目則是在規定 .htaccess 可以覆寫的權限類型有哪些。常見的有：

    - ALL：全部的權限均可被覆寫；
    - AuthConfig：僅有網頁認證 (帳號密碼) 可覆寫；
    - Indexes：僅允許 Indexes 方面的覆寫；
    - Limits：允許使用者利用 Allow, Deny 與 Order 管理可瀏覽的權限；
    - None：不可覆寫，亦即讓 .htaccess 檔案失效！ 這部份我們在進階設定時會再講到的！

- Order, Allow, Deny (能否登入瀏覽的權限)：

  決定此目錄是否可被 apache 的 PID 所瀏覽的權限設定啦！能否被瀏覽主要有兩種判定的方式：

    - deny,allow：以 deny 優先處理，但沒有寫入規則的則預設爲 allow 喔。
    - allow,deny：以 allow 爲優先處理，但沒有寫入規則的則預設爲 deny 喔。 所以在預設的環境中，因爲是 allow,deny 所以預設爲 deny (不可瀏覽)，不過在下一行有個 Allow from all，allow 優先處理，因此全部 (all) 用户端皆可瀏覽啦！這部份我們會在 20.3.4 進階安全設定當中再提及滴。

除了這些資料之外，跟網站資料相關性高的還有底下的幾個咚咚：

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
DirectoryIndex index.html index.html.var  &lt;==首頁『檔案的檔名』設定！
```

如果用户端在網址列只輸入到目錄，例如 http://localhost/ 時，那麼 Apache 將拿出那個檔案來顯示呢？就是拿出首頁檔案嘛！ 這個檔案的檔名在 Apache 當中預設是以 index. 爲開頭的，但 *Windows* 則以 *default.* 之類的檔名爲開頭的。如果你想要讓類似 index.pl 或 index.cgi

也可以是首頁的檔名，那可以改成：

- DirectoryIndex index.html index.htm index.cgi index.pl ...

那如果上面的檔名通通存在的話，那該怎辦？就按照順序啊！接在 DirectoryIndex 後面的檔名參數，越前面的越優先讀取。 那如果檔名通通不存在呢？就是說沒有首頁時，該如何讀取？這就與剛剛談到的 Options 裡面的 Indexes 有關喔！ 這樣有沒有將兩個參數串起來？

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
# Alias   網址列延伸   實際Linux目錄
Alias /icons/ "/var/www/icons/"  &lt;==製作一個目錄別名 (相當類似捷徑)！
&lt;Directory "/var/www/icons"&gt;
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
&lt;/Directory&gt;
```

這個 Alias 很有趣的！是製作出類似連結檔的東西啦！當你輸入 http://localhost/icons 時，其實你的 /var/www/html 並沒有 icons 那個目錄，不過由於 Alias (別名) 的關係，會讓該網址直接連結到 /var/www/icons/ 下。 這裡面預設有很多 Apache 提供的小圖示喔！而因爲設定了一個新的可瀏覽目錄，所以你瞧，多了個 <Directory> 來規定權限了吧！ ^_^

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
# ScriptAlias   網址列延伸   實際Linux目錄
ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"
&lt;Directory "/var/www/cgi-bin"&gt;
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
&lt;/Directory&gt;
```

與上面的 icons 類似，不過這邊卻是以 ScriptAlias (可執行腳本的別名) 爲設定值！ 這個設定值可以指定該目錄底下爲『具有 ExecCGI 』能力的目錄所在喔！所以你可以將類似 Open webmail 的程式給他放置到 /var/www/cgi-bin 內，就不必額外設定其他的目錄來放置你的 CGI 程式喔！ 這樣大概就 OK 了啦！接下來準備一下看看還有哪些額外的設定檔需要處理的呢？

20.2.3 PHP 的預設參數修改

我們前面稍微提過 PHP 是 Apache 當中的一個模組，那在談了 Apache 的 httpd.conf 之後，『我們怎麼沒有講到 PHP 這個模組的設定啊？』不是不講啦！而是因爲目前 Apache 很聰明的將一些重要模組給他拆出來放置到 /etc/httpd/conf.d/*.conf 檔案中了，所以我們必須要到該目錄下才能瞭解到某些模組是否有被加入啊！底下先來瞧瞧吧！

```
[root@www ~]# cd /etc/httpd/conf.d
[root@www conf.d]# ll *.conf
-rw-r--r--. 1 root root 674 Jun 25 15:30 php.conf      &lt;==提供 PHP 模組的設定
-rw-r--r--. 1 root root 299 May 21  2009 welcome.conf  &lt;==提供預設的首頁歡迎訊息
# 如果你是按照剛剛鳥哥說的幾個模組去安裝的，那麼這個目錄下至少會有這兩個資料，
# 一個是規範 PHP 設定，一個則是規範『如果首頁不存在時的歡迎畫面』囉。
```

我們主要來看看關於 PHP 的設定檔吧：

```
[root@www conf.d]# vim /etc/httpd/conf.d/php.conf
&lt;IfModule prefork.c&gt;  &lt;==根據不同的 PID 模式給予不同的 PHP 運作模組
  LoadModule php5_module modules/libphp5.so
&lt;/IfModule&gt;
&lt;IfModule worker.c&gt;
  LoadModule php5_module modules/libphp5-zts.so
&lt;/IfModule&gt;
AddHandler php5-script .php  &lt;==所以副檔名一定要是 .php 結尾！
AddType text/html .php       &lt;==.php 結尾的檔案是純文字檔
DirectoryIndex index.php     &lt;==首頁檔名增加 index.php 喔！
#AddType application/x-httpd-php-source .phps &lt;==特殊的用法！
```

CentOS 6.x 使用的是 PHP 5.x 版本，這個版本依據不同的 apache 使用記憶體模式 (prefork 或 worker) 給予不同的模組！此外，爲了規範 PHP 檔案，因此多了最後三行，包括增加副檔名爲 .php 的檔案處理方式， .php 定義爲純文字檔，以及首頁檔名增加 index.php 等。基本上，這個檔案你不需要有任何的修改，保留原樣即可。

- PHP 的資安方面設定

你必須要知道 PHP 的設定檔其實是在 /etc/php.ini ，這個檔案內容有某些地方可以進行一些小修改， 也有某些地方你必須要特別留意，免得被用户端誤用你的 PHP 資源。底下先介紹一下 PHP 常見的與資安方面較相關的設定：

```
[root@www ~]# vim /etc/php.ini
register_globals = Off
# 這個項目請確定爲 Off (預設就是 Off)，因爲如果設定爲 On 時，
# 雖然程式執行比較不容易出狀況，但是很容易不小心就被攻擊。

log_errors = On
ignore_repeated_errors = On  &lt;==這個設定值調整一下 (因預設爲 Off)
ignore_repeated_source = On  &lt;==這個設定值調整一下
# 這三個設定項目可以決定是否要將 PHP 程式的錯誤記錄起來，
# 建議將重複的錯誤資料忽略掉，否則在很忙碌的系統上，
# 這些錯誤資料將可能造成你的登錄檔暴增，導致效能不佳 (或當機)

display_errors = Off
display_startup_errors = Off
# 當你的程式發生問題時，是否要在瀏覽器上頭顯示相關的錯誤訊息 (包括部分程式碼)
# 強烈的建議設定爲 Off 。不過如果是尚未開放的 WWW 伺服器，爲了你的 debug
# 容易，可以暫時的將他設定爲 On ，如此一來你的程式問題會在瀏覽器上面
# 直接顯示出來，你不需要進入 /var/log/httpd/error_log 登錄當中查閱。
# 但程式完成後，記得將此設定值改爲 Off 喔！重要重要！
```

如果你想要提供 Apache 的說明文件給自己的 WWW 伺服器的話，可以安裝一下 httpd-manual 這個軟體，你就會發現在這個目錄當中又會新增檔案 (manual.conf)，而且從此你可以使用 http://localhost/manual 來登入 Apache 的使用手册呢！眞方便！有興趣的話可以參考

與安裝底下這些軟體喔：

- httpd-manual：提供 Apache 參考文件的一個軟體；
- mrtg：利用類似繪圖軟體自動產生主機流量圖表的軟體；
- mod_perl：讓你的 WWW 伺服器支援 perl 寫的網頁程式(例如 webmail 程式)；
- mod_python：讓你的 WWW 伺服器支援 python 寫的網頁程式。
- mod_ssl：讓你的 WWW 可以支援 https 這種加密過後的傳輸模式。

perl 與 python 是與 PHP 類似的咚咚，都是一些很常用在網頁的程式語言！ 例如知名的 OpenWebMail (http://openwebmail.org/) 就是利用 perl 寫成的。要讓你的 WWW 支援該程式語言， 你就得要安裝這些東西啦！(但不是所有的軟體都安裝！請安裝你需要的即可！)

- PHP 提供的上傳容量限制

我們未來可能會使用 PHP 寫成的軟體來提供用户上傳/下載檔案資料，那麼 PHP 有沒有限制檔案容量呢？ 答案是有的！那麼容量限制是多大？預設是 2M 左右。你可以修改它的，假設我們現在要限制成爲 16MBytes 時，我們可以這樣修訂：

```
[root@www ~]# vim /etc/php.ini
post_max_size = 20M       &lt;==大約在 729 行左右
file_uploads = On         &lt;==一定要是 On 才行 (預設值)
upload_max_filesize = 16M &lt;==大約在 878 行左右
memory_limit = 128M       &lt;==PHP 可用記憶體容量也能修訂！
```

與檔案上傳/下載容量較相關的就是這幾個設定值～爲啥 post_max_size 要比 upload_max_filesize 大呢？ 因爲檔案有可能也是透過 POST 的方式傳輸到我們伺服器上頭，此時你的檔案就得要加入 POST 資訊內～ 因爲 POST 資訊可能還含有其它的額外資訊，所以當然要比檔案容量大才行！所以在設計這個設定檔時， 這兩個值得要特別注意喔！

20.2.4 啓動 WWW 服務與測試 PHP 模組

OK！最單純簡易的 WWW 伺服器設定搞定的差不多了，接下來就是要啓動啦！啓動的方法簡單到不行，用傳統的方式來處理：

```
[root@www ~]# /etc/init.d/httpd start       &lt;==立刻啓動啦！
[root@www ~]# /etc/init.d/httpd configtest  &lt;==測試設定檔語法
[root@www ~]# chkconfig httpd on            &lt;==開機啓動 WWW 啦！
```

另外，其實 Apache 也自行提供一支 script 可以讓我們來簡單的使用，那就是 apachectl 這支程式啦！ 這支程式的用法與 /etc/init.d/httpd 幾乎完全一模一樣喔！

```
[root@www ~]# /usr/sbin/apachectl start  &lt;==啓動啦！
[root@www ~]# /usr/sbin/apachectl stop   &lt;==關閉 WWW 啦！
```

一般建議你可以稍微記一下 apachectl 這支程式，因爲很多認證考試會考，而且他也是 Apache 預設提供的一個管理指令説！好了，來看看有沒有啓動成功？

```
# 先看看 port 有沒有啓動啊！
[root@www ~]# netstat -tulnp &#124; grep 'httpd'
Proto Recv-Q Send-Q Local Address  Foreign Address State  PID/Program name
tcp        0      0 :::80          :::*            LISTEN 2493/httpd

# 再來看看登錄檔的資訊記錄了什麼！這個確實建議瞧一瞧！
[root@www ~]# tail /var/log/httpd/error_log
[notice] &lt;u&gt;SELinux policy enabled; httpd running as context&lt;/u&gt; unconfined_u
[notice] suEXEC mechanism enabled (wrapper: /usr/sbin/suexec)
[notice] Digest: generating secret for digest authentication ...
[notice] Digest: done
[notice] &lt;u&gt;Apache/2.2.15 (Unix) DAV/2 PHP/5.3.2 configured -- resuming normal oper
# 第一行在告知有使用 SELinux(強調一下)，最後一行代表正常啓動了！
```

這樣應該就成功啓動了 Apache 囉！比較重要的是還有啓動 SELinux 的相關說明，這底下我們還得要注意注意呢！ 接下來測試看看能不能看到網頁呢？首先看看 /var/www/html 有沒有資料？咦？沒有～沒關係，因爲 CentOS 幫我們造了一個測試頁了 (Apache 的 welcome 模組功能)，所以你還是在瀏覽器上面輸入你這部主機的 IP 看看先：

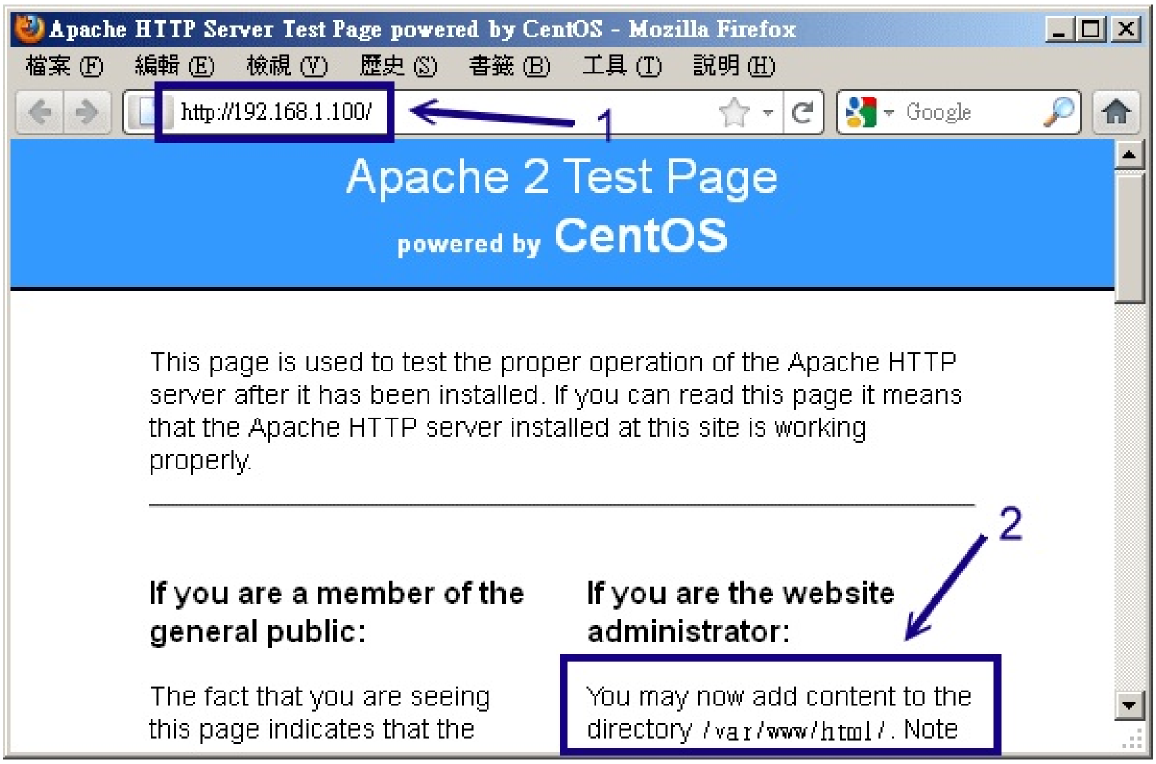


圖 20.2-1、啓動 Apache 之後，所看到的預設首頁

你可以在伺服器上面啓動圖形介面來查閱，也可以透過客户端電腦來連線 (假設防火牆問題已經克服了)。 鳥哥這裡假設伺服器爲 runlevel 3 的純文字介面，因此使用外部的用户端電腦連線到伺服器的 IP 上，如上圖畫面中的箭頭 1 處。如果你是在伺服器本機上面啓動的瀏覽器，那直接輸入『 http://localhost 』即可。同時看到畫面中的箭頭 2 所指處，你就可以發現首頁的位置是在 /var/www/html/ 底下囉！但如果想要知道有沒有成功的驅動 PHP 模組，那你最好先到 /var/www/html 目錄下去建立一個簡單的檔案：

```
[root@www ~]# vim /var/www/html/phpinfo.php
&lt;?php  phpinfo ();  ?&gt;
```

要記住，PHP 檔案的副檔名一定要是 .php 結尾的才行喔！至於內容中，那個『 <?php ... ?> 』是嵌入在 HTML 檔案內的 PHP 程式語法，在這兩個標籤內的就是 PHP 的程式碼。那麼 phpinfo(); 就是 PHP 程式提供的一個函式庫，這個函式庫可以顯示出你 WWW 伺服器內的相關服務資訊， 包括主要的 Apache 資訊與 PHP 資訊等等。這個檔案建置完畢後，接下來你可以利用瀏覽器去瀏覽一下這個檔案：



圖 20.2-2、測試 Apache 能否驅動 PHP 模組

注意看網址的部分喔！因為我們 phpinfo.php 是放置在首頁目錄底下，因此整個 URL 當然就成為上述箭頭 1 當中的模樣了。 這個 phpinfo() 函數輸出的內容還挺機密的，所以測試完畢後請將這個檔案刪除吧！從上頭的畫面你可以知道 PHP 模組的版本以及 Apache 相關的重要資料啦！自己仔細瞧瞧吧！如此一來，你的 Apache 與 PHP 就 OK 的啦！

那萬一測試失敗怎麼辦？常見的錯誤問題以及解決之道可以參考：

- 網路問題：雖然在本機上沒有問題， 但不代表網路一定是通的！請確認一下網路狀態！例如 Route table, 撥接情況等等；
- 設定檔語法錯誤：這個問題很常發生，因為設定錯誤，導致無法將服務啟動成功。 此時除了參考螢幕上面的輸出資訊外，你也可以透過 /etc/init.d/httpd configtest 測試語法，更佳的解決方案是參考 /var/log/httpd/error_log 內的資料，可以取得更詳盡的解決之道。
- 權限問題：例如你剛剛在 httpd.conf 上面的 user 設定為 apache 了，但偏偏要被瀏覽的檔案或目錄權限對 apache 沒有可讀權限，自然就無法讓人家連線進去啦！
- 問題的解決之道：如果還是沒有辦法連接上來你的 Linux Apache 主機，那麼請：
    1. 察看 /var/log/httpd/error_log 這個檔案吧！他應該可以告訴你很多的資訊喔！

2. 仔細的察看一下你瀏覽器上面顯示的資訊，這樣才能夠知道問題出在哪裡！
3. 另一個可能則是防火牆啦！察看一下 iptables 的訊息！也可能是 SELinux 的問題喔！

20.2.5 MySQL 的基本設定

在 LAMP 伺服器裡面，Linux, Apache, PHP 已經處理完畢，那麼 MySQL 呢？所以，接下來就是要處理這個資料庫軟體囉。 在啟動 MySQL 前其實系統並沒有幫我們建立任何的資料庫。當你初次啟動 MySQL 後，系統才會針對資料庫進行初始化的建立啊。 不相信的話你可以先看看 /var/lib/mysql/ 這個目錄，裡面其實沒有任何資料的啦。

- 啟動 MySQL (設定 MySQL root 密碼與新增 MysQL 用戶帳號)

首先得要啟動 MySQL 才行，啟動的方法還是很簡單啊！

```
[root@www ~]# /etc/init.d/mysqld start
[root@www ~]# chkconfig mysqld on
# 如果是初次啟動，螢幕會顯示一些訊息且 /var/lib/mysql 會建立資料庫。

[root@www ~]# netstat -tulnp &#124; grep 'mysql'
Proto Recv-Q Send-Q Local Address  Foreign Address   State   PID/Program name
tcp        0      0 0.0.0.0:3306   0.0.0.0:*         LISTEN  2726/mysqld

# 底下在測試看能否以手動的方式連上 MySQL 資料庫！
[root@www ~]# mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.1.52 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql&gt; exit
Bye
```

MySQL 預設監聽的埠口在 port 3306，從上面看來我們的 MySQL 似乎是啟動了，不過剛剛初始化的 MySQL 資料庫管理員並沒有任何密碼，所以很可能我們的資料庫是會被使用者搞爛掉的～所以你最好對 MySQL 的管理員帳號設定一下密碼才好。 另外，上面那個 root 與我們 Linux 帳號的 root 是完全無關的！因爲 MySQL 資料庫軟體也是個多人的操作環境，在該軟體內有個管理者恰好帳號也是 root 而已。

那麼如何針對 MySQL 這個軟體內的 root 這個管理者設定他的密碼呢？你可以這樣做：

```
[root@www ~]# mysqladmin -u root password 'your.password'
# 從此以後 MySQL 的 root 帳號就需要密碼了！如下所示：

[root@www ~]# mysql -u root -p
Enter password:  &lt;==你必須要在這裡輸入剛剛建立的密碼！

mysql&gt; exit
```

如此一來 MySQL 資料庫的管理方面會比較安全些啦！其實更好的作法是分別建立不同的使用者管理不同的資料庫。 舉例來說，如果你要給予 vbirduser 這個使用者一個 MySQL 的資料庫使用權，假設你要給他的資料庫名稱爲 vbirddb，且密碼爲 vbirdpw 時，你可以這樣做：

```
[root@www ~]# mysql -u root -p
Enter password:  &lt;==如前所述，你必須要輸入密碼嘛！
mysql&gt; create database vbirddb;  &lt;==注意每個指令後面都要加上分號 (;)
Query OK, 1 row affected (0.01 sec)

mysql&gt; grant all privileges on vbirddb.* to vbirduser@localhost
identified by 'vbirdpw' ;
Query OK, 0 rows affected (0.00 sec)

mysql&gt; show databases;
+--------------------+
&#124; Database           &#124;
+--------------------+
&#124; information_schema &#124;
&#124; mysql              &#124; &lt;==用來記錄 MySQL 帳號、主機等重要資訊的主要資料庫！
&#124; test               &#124;
&#124; vbirddb            &#124; &lt;==我們剛剛建立的資料庫在此
+--------------------+
4 rows in set (0.00 sec)

mysql&gt; use mysql;
mysql&gt; select * from user where user = 'vbirduser';
# 上面兩個指令在查詢系統有沒有 vbirduser 這個帳號，若有出現一堆東西，
# 那就是查詢到該帳號了！這樣就建置妥當囉！

mysql&gt; exit
```

然後你可以利用『 mysql -u vbirduser -p 』這個指令來嘗試登入 MySQL 試看看，嘿嘿！就知道 vbirduser 這個使用者在 MySQL 裡面擁有一個名稱爲 vbirddb 的資料庫啦！其他更多的用法就得請你自行參考 SQL 相關的語法囉！不在本文的討論範圍啦！

- 效能調校 /etc/my.cnf

由於 MySQL 這個資料庫系統如果在很多使用者同時連線時，可能會造成某些效能方面的瓶頸， 因此，如果你的資料庫真的好大好大，建議可以改用 postgresql 這套軟體，這套軟體的使用與 mysql 似乎差異不大。 不過，我們還是提供一些簡單的方式來處理小站的 MySQL 效能好了。相關的資料鳥哥是參考這一篇簡單的中文說明：

- http://parus1974.wordpress.com/2005/02/27/mysql再調整

```
[root@www ~]# vim /etc/my.cnf
[mysqld]
default-storage-engine=innodb
# 關於目錄資料與語系的設定等等；
default-character-set    = utf8   &lt;==每個人的編碼都不相同，不要隨意跟我一樣
port                     = 3306
skip-locking
# 關於記憶體的設定，注意，記憶體的簡單計算方式爲：
# key_buffer + (sort_buffer + read_buffer ) * max_connection
# 且總量不可高於實際的實體記憶體量！所以，我底下的資料應該是 OK 的
# 128 + (2+2)*150 = 728MB
key_buffer               = 128M
sort_buffer_size         = 2M
read_buffer_size         = 2M
join_buffer_size         = 2M
max_connections          = 150
max_connect_errors       = 10
read_rnd_buffer_size     = 4M
max_allowed_packet       = 4M
table_cache              = 1024
myisam_sort_buffer_size = 32M
thread_cache             = 16
query_cache_size         = 16M
tmp_table_size           = 64M
# 由連線到確定斷線的時間，原本是 28800 (sec) ，約 8 小時，我將他改爲 20 分鐘！
wait_timeout             = 1200
thread_concurrency       = 8
innodb_data_file_path = ibdata1:10M:autoextend
innodb_buffer_pool_size = 128M
innodb_additional_mem_pool_size = 32M
innodb_thread_concurrency = 16

datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
user=mysql
symbolic-links=0
[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
```

你要注意的是，因爲鳥哥的主機上面假設記憶體有 2GB 啊！所以跟記憶體相關的資料才會寫很大！ 請依照你實際擁有的記憶體量來處理喔！還得加上你的 Apache 本身的記憶體用量！ 所以...如果你的網站流量很大的話，在校能測試上面要很注意啊！

- MySQL root 密碼忘記的緊急處理

如果你不小心忘記 MySQL 的密碼怎麼辦？網路上有一些工具可以讓你去處理 MySQL 資料庫的挽回。 如果你的資料庫內容並不是很重要，刪除也無所謂的話 (測試中 @_@)，那麼可以將 MySQL 關閉後， 將 /var/lib/mysql/* 那個目錄內的資料刪除掉，然後再重新啓動 MySQL ，那麼 MySQL 資料庫會重建， 你的 root 又沒有密碼啦！

不過，這個方法僅適合你的資料庫並不重要的時候，如果資料庫很重要...那千萬不要隨便刪除啊！

20.2.6 防火牆設定與 SELinux 的規則放行

設定好了 LAMP 之後，開始要讓用户端來連線啊！那麼如何放行呢？要放行哪些埠口？剛剛的 port 3306 要不要放行？ 這裡請注意，如果是小型的 WWW 網站，事實上，Apache 是連接本機的 MySQL，並沒有開放給外部的用户來連接資料庫！ 因此，請不要將 3306 放行給網際網路連接，除非你真的知道你要給其他的伺服器讀取你的 MySQL 喔！既然如此，當然只要開放 port 80 即可。

此外，如果你的 Apache 未來還想要進行一些額外的連線工作，那麼 SELinux 的一些簡單規則也得先放行！否則會有問題啦！不過 SELinux 的問題其實都好解決，因爲可以參考登錄檔來修訂嘛！ 好了，讓我們簡單的來談談：

```
# 1\. 放行防火牆中的 port 80 連線
[root@www ~]# vim /usr/local/virus/iptables/iptables.rule
iptables -A INPUT -p TCP -i $EXTIF --dport  80  --sport 1024:65534 -j ACCEPT
# 將上面這一行的註解拿掉即可！
[root@www ~]# /usr/local/virus/iptables/iptables.rule
[root@www ~]# iptables-save &#124; grep 80
-A PREROUTING -s 192.168.100.0/255.255.255.0 -i eth1 -p tcp -m tcp --dport 80
   -j REDIRECT --to-ports 3128 &lt;==這一行是進行 squid 產生的，應該要拿掉較佳
-A INPUT -i eth0 -p tcp -m tcp --sport 1024:65534 --dport 80 -j ACCEPT
# 看到上面這行，就是將防火牆的放行加進來了，用户端應該是能夠連線囉！

# 2\. 解決 SELinux 的規則放行問題：
[root@www ~]# getsebool -a &#124; grep httpd  &lt;==會出現一堆規則，有興趣的如下：
[root@www ~]# setsebool -P httpd_can_network_connect=1
# 其他的規則或類型，等待後續的章節介紹再來談！
```

例題：你想要修改首頁內容，且先使用 root 在 /root 底下建立了 index.html 了，這個檔案將被移動到 /var/www/html 底下， 請建立該檔案，並且放置成首頁檔案，瀏覽看看。答：可以透過簡單的方式建立一個無關緊要的首頁檔案：

```
[root@www ~]# echo "This is my Home page" &gt; index.html
[root@www ~]# mv index.html /var/www/html
[root@www ~]# ll /var/www/html/index.html
-rw-r--r--. 1 root root 21 2011-08-08 13:49 /var/www/html/index.html
# 權限看起來是 OK 的！
```

現在請使用瀏覽器瀏覽一下 http://localhost ，就會發現無法讀取！爲什麼？請檢查 /var/log/httpd/error_log 以及 /var/log/messages 的內容：

```
[root@www ~]# tail /var/log/httpd/error_log
[error] [client 192.168.1.101] (13)Permission denied: access to /index.html denied
[root@www ~]# tail /var/log/messages
Aug  8 13:50:14 www setroubleshoot: SELinux is preventing /usr/sbin/httpd "getattr"
 access to /var/www/html/index.html. For complete SELinux messages. &lt;u&gt;run sealert
 6c927892-2469-4fcc-8568-949da0b4cf8d&lt;/u&gt;
```

看到上面畫底下的地方了吧？就是他！執行一下，你就能發現如何處理囉！

20.2.7 開始網頁設計及安裝架站軟體，如 phpBB3

基礎的 LAMP 伺服器架設完畢之後，基本上，你就可以開始設計你想要的網站囉！編寫網頁的工具很多，請自行尋找吧！ 不過對於這個簡單的 LAMP 伺服器，你必須要知道的是：

- 預設的首頁目錄在 /var/www/html/，你應該將所有的 WWW 資料都搬到該目錄底下才對！
- 注意你的資料權限 (rwx 與 SELinux)！務必要讓 Apache 的程序使用者能夠瀏覽！
- 盡量將你的首頁檔案檔名取爲 index.html 或 index.php ！
- 如果首頁想要建立在其他地方，你應該要修改 DocumentRoot 那個參數 (httpd.conf)
- 不要將重要資料或者隱私資料放置到 /var/www/html/ 首頁內！
- 如果你需要安裝一些 CGI 程式的話，建議你將他安裝到 /var/www/cgi-bin/ 底下， 如此一來你不需要額外設定 httpd.conf 即可順利啓動 CGI 程式；

除了這些基本的項目之外，其實你可以使用網際網路上面人家已經做好的 PHP 程式架站機喔！ 譬如說討論區軟體 phpBB3 這個玩意兒，完整的架站軟體 PHPNuke 以及部落格軟體 lifetype 等等。 但這些架站機都需要 PHP 與資料庫的支援，所以你必需要將上述介紹的 LAMP 完整的安裝好才行。 如果你不喜歡自己寫網頁的話，那麼這些有用的架站軟體就夠你瞧的囉！鳥哥列出幾個連結給你玩一玩先！

- phpBB 討論區官方網站：http://www.phpbb.com/
- phpBB 正體中文網站『竹貓星球』：http://phpbb-tw.net
- 鳥哥的簡易 phpBB 安裝法：http://linux.vbird.org/apache_packages/
- Lifetype 部落格架設軟體中文支援站：http://www.lifetype.org.tw/
- Lifetype 部落格架設軟體官網：http://www.lifetype.net/
- PHP-Nuke 官方網站：http://phpnuke.org/
- xoops 官方網站：http://www.xoops.org/

不過請注意，這些軟體由於是公開的，所以有些怪叔叔可能會據以亂用或亂改，因此可能會有一些 bug 會出現！因此，你必需要取得最新的版本來玩才行，而且架設之後還得要持續的觀察是否有更新的版本出現， 隨時去更新到最新版本才行喔！免得後患無窮～

20.3 Apache 伺服器的進階設定

事實上，剛剛上頭的基本設定已經很足夠朋友們架設 WWW 伺服器所需了！不過，還有很多可以玩玩的地方，例如個人用户首頁、虛擬主機以及認證保護的網頁等等。底下我們分別來談一談囉！

20.3.1 啓動用户的個人網站 (權限是重點)

每一部 WWW 伺服器都有一個首頁，但是如果每個個人用户都想要有可以自己完全控管的首頁時，那該如何設計？呵呵！Apache 早就幫我們想到了！不過新版的設定檔內常常是預設將這個功能取消的，所以你必需要自行修訂呢！

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
# 找到如下的設定項目，大約在 366 行左右：
&lt;IfModule mod_userdir.c&gt;
    UserDir disable
    #UserDir public_html
&lt;/IfModule&gt;
# 將他改成如下的情況喔！
&lt;IfModule mod_userdir.c&gt;
    #UserDir disable
    UserDir www
&lt;/IfModule&gt;

# 重新啓動一下先！
[root@www ~]# /etc/init.d/httpd restart
```

這只是個範例，Apache 預設的個人首頁是放置在家目錄下的 ~/public_html/ 目錄下！假如你的系統有個帳號叫做 student， 那麼預設的屬於 student 的個人首頁就會放置在 /home/student/public_html/ 底下。不過，這個 public_html 實在很討厭， 看起來跟網頁沒有什麼特殊關連性，因此鳥哥都會將這個目錄改爲 www，所以 student 的個人首頁就會是在 /home/student/www/ 目錄下，比較好記憶。

例題：如何讓未來所有『新增』的使用者預設家目錄下都有個 www 的目錄？ 答：因爲新增使用者時所參考的家目錄在 /etc/skel 目錄內，所以你可以直接 mkdir /etc/skel/www 即可。 若想要讓用户直接擁有一個簡易的首頁，還能夠使用 echo "My homepage" > /etc/skel/www/index.html 呢！

- 個人首頁的 URL 以及目錄的權限、SELinux 設定

現在假設我們要讓已經存在系統中的 student 這個帳號具有個人首頁，那就得要手動去建置所需要的目錄與檔案才行。 現在請登入 student，並用該帳號建置底下的相關資訊：

```
[student@www ~]$ mkdir www
[student@www ~]$ chmod 755 www   &lt;==針對 www 目錄開放權限
[student@www ~]$ chmod 711 ~     &lt;==不要忘了家目錄也要改！
[student@www ~]$ cd www
[student@www www]$ echo "Test your home" &gt;&gt; index.html
```

由於 CentOS 預設的使用者家目錄權限是 drwx------ ，這個權限將無法讓 Apache 的程序瀏覽啊！所以你至少要讓你的家目錄權限成爲 drwx--x--x 才行！這個很重要啊！ 那麼未來只要你在瀏覽器的網址列這樣輸入：

- http://你的主機名稱/~student/

『理論上』就能夠看到你的個人首頁了。不過，可惜的是，我們的 SELinux 並沒有放行個人首頁！所以，此時你會發現瀏覽器出現 『You don't have permission』的訊息！趕緊看一下你的 /var/log/messages，裡面應該會教你進行這項工作：

```
[root@www ~]# setsebool -P httpd_enable_homedirs=1
[root@www ~]# restorecon -Rv /home/
# 第一個指令在放行個人首頁規則，第二個指令在處理安全類型！
```

就可以看到你的使用者個人網頁囉！之後讓使用者自己去設計他的網站吧！現在你知道那個毛毛蟲 (~) 在 URL 上面的意義了吧？ ^_^！不過，多這個毛毛蟲就很討厭～我可不可以將使用者的個人網站設定成爲：

- http://你的主機名稱/student/

是可以啦！最簡單的方法是這樣的：

```
[root@www ~]# cd /var/www/html
[root@www html]# ln -s /home/student/www student
```

由於我們首頁的『 Options 』內有 FollowSymLinks 這個參數的原因，所以可以直接使用連結檔即可。 另外我們也可以使用 Apache 提供的別名功能 (Alias)，例如這樣做：

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
# 找個不與人家設定值有干擾的地方加入這個設定項目：
Alias /student/ "/home/student/www/"
&lt;Directory "/home/student/www"&gt;
        Options FollowSymLinks
        AllowOverride None
        Order allow,deny
        Allow from all
&lt;/Directory&gt;

[root@www ~]# /etc/init.d/httpd restart
```

不過，如果你使用這個方法的話得要特別注意，在 httpd.conf 內的 Alias 後面接的目錄，需要加上目錄符號 (/) 在結尾處， 同時，網址列必須要輸入 http://IP/student/ ！亦即是結尾也必須要加上斜線才行！否則會顯示找不到該 URL 喔！

20.3.2 啓動某個目錄的 CGI (perl) 程式執行權限

在前幾個小節裡面我們有談到，如果你想要 Apache 可以執行 perl 之類的網頁程式時， 你就得需要安裝一些額外的模組才行。其中 mod_perl 與 mod_python 這兩個軟體建議你最好安裝一下啦！然後我們也提到想要執行 CGI 程式就得到 /var/www/cgi-bin/ 目錄下去執行。如果你想要在其他目錄底下執行 CGI 程式是否可以？當然行啊！

- 利用新目錄下的 Options 參數設定：

假設想要執行 CGI 的程式附檔名爲 .cgi 或 .pl ，且放置的目錄在 /var/www/html/cgi/ 時，你可以這樣做：

```
[root@www ~]# yum install mod_python mod_perl
[root@www ~]# vim /etc/httpd/conf/httpd.conf
# 找到底下這一行，大約在 797 行左右啦：
#AddHandler cgi-script .cgi
# 將他改成底下的模樣，讓附檔名爲 .pl 的檔案也能執行喔！
AddHandler cgi-script .cgi .pl

# 然後加入底下這幾行來決定開放某個目錄的 CGI 執行權限。
&lt;Directory "/var/www/html/cgi"&gt;
    Options +ExecCGI
    AllowOverride None
    Order allow,deny
    Allow from all
&lt;/Directory&gt;

[root@www ~]# /etc/init.d/httpd restart
```

接下來只要讓你的 CGI 程式具有 x 權限，那麼他就可以執行啦！舉例來說，你的檔案在 /var/www/html/cgi/helloworld.pl 的話，那麼：

```
[root@www ~]# mkdir /var/www/html/cgi
[root@www ~]# vim /var/www/html/cgi/helloworld.pl
#!/usr/bin/perl
print "Content-type: text/html\r\n\r\n";
print "Hello, World.";
[root@www ~]# chmod a+x /var/www/html/cgi/helloworld.pl
```

然後在網址列輸入：『http://主機名稱或 IP/cgi/helloworld.pl』即可執行該檔案並將結果顯示在螢幕上面囉！

- 使用 ScriptAlias 的功能：

你可以直接利用檔名的別名來處理即可！更簡單呢。我們現在假設所有在 /var/www/perl/ 目錄下的檔案都可以是 perl 所撰寫的程式碼，那麼我們可以這樣做：

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
# 同樣的你要先確認這一行是存在的！
AddHandler cgi-script .cgi .pl

# 然後加入底下這幾行來決定開放某個目錄的 CGI 執行權限。
ScriptAlias /perl/ "/var/www/perl/"

[root@www ~]# /etc/init.d/httpd restart

[root@www ~]# mkdir /var/www/perl
[root@www ~]# cp -a /var/www/html/cgi/helloworld.pl /var/www/perl
```

現在，請在網址列輸入：『http://IP/perl/helloworld.pl』，就能夠看到剛剛的資料了！這個方法比較棒啦！因爲該目錄不需要在 Apache 首頁底下也可以成功的啦！這兩個方法你可以隨意取一個來處理即可！不需要兩個都進行啦！

20.3.3 找不到網頁時的顯示訊息通知

如果你的 /var/www/html/cgi 目錄底下沒有任何首頁檔案 (index.???) 時，那當使用者在網址列輸入『 http://your.hostname/cgi 』，請問結果會顯示出什麼呢？可能有兩個：

- 如果你的 Options 裡面有設定 Indexes 的話，那麼該目錄下的所有檔案都會被列出來，提供類似 FTP 的連結頁面。
- 如果沒有指定 Indexes 的話，那麼錯誤訊息通知就會被顯示出來。

事實上 CentOS 所提供的 Apache 已經規範好一些簡單的錯誤資料網頁了，你可以到 /var/www/error/ 目錄下瞧瞧就曉得。不過該目錄下的檔案並沒有中文訊息，所以...眞要命！至於 Apache 的錯誤訊息設定在這裡：

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
# 大約在 875 行左右，預設就是註解掉的！
#    ErrorDocument 403 /error/HTTP_FORBIDDEN.html.var
#    ErrorDocument 404 /error/HTTP_NOT_FOUND.html.var
#    ErrorDocument 405 /error/HTTP_METHOD_NOT_ALLOWED.html.var
#    ErrorDocument 408 /error/HTTP_REQUEST_TIME_OUT.html.var
....(後面省略)....
```

雖然 Apache 預設有提供一些額外的資料給我們使用，不過，鳥哥不是很喜歡那樣的畫面啦！反而比較喜歡像是 Yahoo 或是其他大型的網站所提供的資訊頁面，可以提供給用户一些有效的連結，這樣會比較方便用户連結到我們的網站啊！ 此時我們可以這樣做：

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
# 找到底下這一段，大約在 836 行左右，看看這些簡單的範例先：
#ErrorDocument 500 "The server made a boo boo."
ErrorDocument 404 /missing.html  &lt;==將註解拿掉吧！
#ErrorDocument 404 "/cgi-bin/missing_handler.pl"
#ErrorDocument 402 http://www.example.com/subscription_info.html

[root@www ~]# /etc/init.d/httpd restart
```

上面那個檔案 /missing.html 必需要放置在你的首頁目錄下，亦即是 /var/www/html/missing.html 啦！ 要提醒你的是：『你的所有設定檔當中 (包括 /etc/httpd/conf.d/*.conf) 只能存在一個 ErrorDocument 404 ... 的設定值，否則將以較晚出現的設定爲主』。 所以你得先搜尋一下，尤其是很多 Linux 版本的 Apache 並沒有將預設的錯誤訊息註解呢。至於那個 404 是啥意思？他的意義是這樣的：

- 100-199：一些基本的訊息
- 200-299：用戶端的要求已成功的達成
- 300-399：Client 的需求需要其他額外的動作，例如 redirected 等等
- 400-499：Client 的要求沒有辦法完成(例如找不到網頁)
- 500-599：主機的設定錯誤問題

好了，接下來讓我們編輯一下那個 missing.html 的檔案內容吧！ ^_^

```
[root@www ~]# vim /var/www/html/missing.html
&lt;html&gt;
&lt;head&gt;
        &lt;meta http-equiv="Content-Type" content="text/html; charset=utf8"&gt;
        &lt;title&gt;錯誤訊息通知&lt;/title&gt;
&lt;head&gt;
&lt;body&gt;
        您輸入的網頁找不到！&lt;br /&gt;
        &lt;hr /&gt;
        親愛的網友，你所輸入的網址並不存在我們的伺服器當中，
        有可能是因爲該網頁已經被管理原刪除，
        或者是你輸入了錯誤的網址。請再次查明後在填入網址囉！
        或按&lt;a href="/"&gt;這裡&lt;/a&gt;回到首頁。
        感謝你常常來玩！ ^_^&lt;br /&gt;
        &lt;hr /&gt;
        若有任何問題，歡迎聯絡管理員&lt;a
        href="mailto:vbird@www.centos.vbird"&gt;vbird@www.centos.vbird&lt;/a&gt;。
&lt;/body&gt;
&lt;/html&gt;
```

現在你如果在網址列隨便輸入一個伺服器上不存在的網址，就會出現如下的畫面囉：

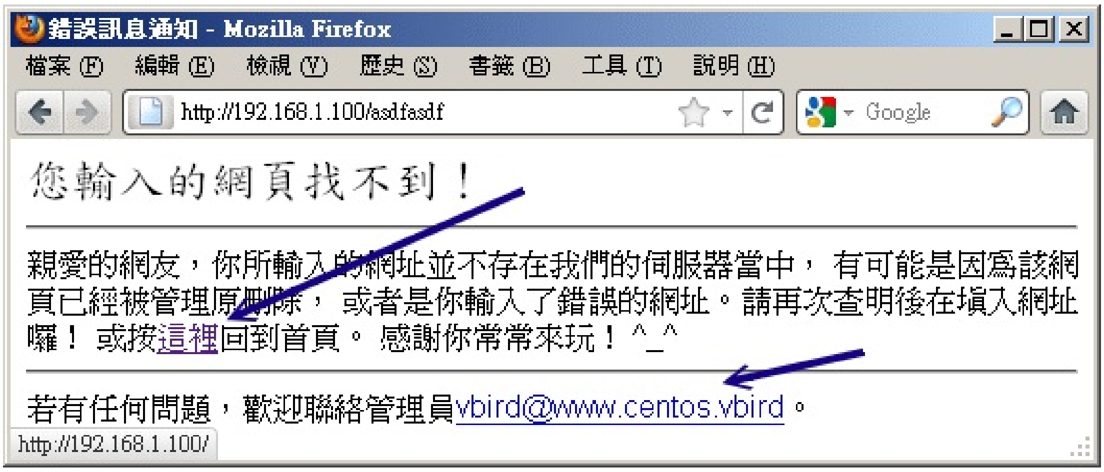


圖 20.3-1、找不到網頁時的錯誤通報訊息

當然啦，你可以自行設定出符合你網頁風格的資料啦！例如鳥哥的網站上面就列出一些基本的連結， 幫助網友們可以順利的取得他們想要的資料啊！這也是很重要的功能呢！^_^

20.3.4 瀏覽權限的設定動作 (order, limit)

你該如何限制用户端對你的 WWW 連線呢？你會説，那就利用 iptables 這個防火牆嘛！那有什麼難的？問題是， 如果同一個 IP 來源，他某些網頁可以瀏覽，但某些網頁不能瀏覽時，該如何設定？iptables 僅能一口氣開放或整個拒絕， 無法針對 WWW 的內容來部分放行。那該如何處理？就透過 apache 內建的 order 項目來處置即可。 先來回憶一下 order 搭配 allow, deny 的相關限制：

- Order deny,allow：以 deny 優先處理，但沒有寫入規則的則預設爲 allow。常用於：拒絕所有，開放特定的條件；
- Order allow,deny：以 allow 爲優先處理，但沒有寫入規則的則預設爲 deny。常用於：開放所有，拒絕特定的條件。
- 如果 allow 與 deny 的規則當中有重複的，則以預設的情況 (Order 的規範) 爲主。

舉例來説，如果我們的首頁目錄想要讓 192.168.1.101 及政府部門無法連線，其他的則可以連線， 由上面的説明你可以知道這是『開放所有，拒絕特定』的條件，所以你可以這樣做設定：

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
&lt;Directory "/var/www/html"&gt;
    Options FollowSymLinks
    AllowOverride None
    Order allow,deny
    Allow from all
 Deny from 192.168.1.101  &lt;==約在 344 行新增底下兩行！
    Deny from .gov.tw
&lt;/Directory&gt;

[root@www ~]# /etc/init.d/httpd restart
```

注意一下，因爲 Order 是『 allow,deny 』，所以所有規則當中屬於 allow 的都會被優先提到最上方， 爲了避免這個設計上的困擾，所以建議你直接將 allow 的規則寫在最上方。 而由於規則當中 192.168.1.101 隸屬於 all 當中 (all 代表所有的嘛！)，因此這個設定項目則爲預設值， 亦即爲 deny 啦！那個 .gov.tw 的設定項目也一樣。如果是底下的模樣：

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
# 底下可是個錯誤的示範，請仔細看下個段落的詳細説明喔！
&lt;Directory "/var/www/html"&gt;
    Options FollowSymLinks
    AllowOverride None
 Order deny,allow
    Deny from 192.168.1.101
    Deny from .gov.tw
    Allow from all
&lt;/Directory&gt;
```

雖然 deny 會先挪到上方來處理，不過因爲 192.168.1.101 是在 all 的範圍內，所以發生重複， 因此這個設定值將會以預設的 allow 爲主，因此就無法限制住這個 192.168.1.101 的存取啦！ 這邊很容易搞錯的吶！鳥哥也是常常搞到頭昏腦脹的～

例題：如果有個應該要保護的內部目錄，假設在 /var/www/html/lan/ ， 我僅要讓 192.168.1.0/24 這個網域可以瀏覽的話，那麼你應該要如何設定的好？答：這個案例當中有點像是『拒絕所有連線，僅接受特定連線』的樣子，因此可以使用 deny,allow 那個情況， 所以你可以這樣做：

```
&lt;Directory "/var/www/html/lan"&gt;
    Options FollowSymLinks
    AllowOverride None
    Order deny,allow
    deny from all
    allow from 192.168.1.0/24
&lt;/Directory&gt;
```

事實上，如果想要讓某個網域或者是 IP 無法瀏覽的話，最好還是利用 iptables 來處理比較妥當。 不過如果僅是某些重要目錄不想讓人家來查閱的話，那麼這個 allow, deny 與 order 的設定資料可就很值得參考了。

而除了這個 order 設定值之外，我們還有個限制用户端能進行的動作的設定喔！ 那就是 Limit 這個設定啦！舉例來說，如果我們想要讓使用者在 /var/www/html/lan 這個目錄下僅能進行最陽春的 GET, POST, OPTIONS 的功能，除了這幾個之外的其他功能通通不允許， 那麼你可以這樣做：

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
&lt;Directory "/var/www/html/lan"&gt;
    AllowOverride none
    Options FllowSymLinks
 # 先允許能夠進行 GET, POST 與 OPTIONS 啦！
    &lt;Limit GET POST OPTIONS&gt;
        Order allow,deny
        Allow from all
    &lt;/Limit&gt;

    # 再規定除了這三個動作之外，其他的動作通通不允許啦！
    &lt;LimitExcept GET POST OPTIONS&gt;
        Order deny,allow
        Deny from all
    &lt;/LimitExcept&gt;
&lt;/Directory&gt;
```

透過 Limit 與 LimitExcept 就能夠處理用户端能夠進行的動作啦！也就有辦法針對你的資料進行細部保護囉。 不過這些保護真的很細部，一般小網站大致上用不到 Limit 這個玩意兒說。

20.3.5 伺服器狀態說明網頁

既然已經安裝好了 WWW 伺服器，除了提供服務之外，重要的是要如何維護囉！嘿嘿！ 那麼是否一定要額外安裝其他的軟體才能知道目前的主機狀態呢？當然不需要啦！我們可以透過 Apache 提供的特別功能來查詢主機目前的狀態！那就是 mod_status 這個模組囉！ 這個模組預設是關閉的，你必須要修改設定檔來啓動他才行。

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
# 先確定底下這幾個項目真的有存在才行！
LoadModule status_module modules/mod_status.so &lt;==大約在 178 行，就是模組的載入

ExtendedStatus On  &lt;==大約在 228 行，你可以將他打開，資訊會比較多！

# 底下的資料則大約在 924 行左右，你可以將他修改成爲這樣：
&lt;Location /server-status&gt;
    SetHandler server-status
    Order deny,allow
    Deny from all
    Allow from 192.168.1.0/24
    Allow from 127.0.0.1
&lt;/Location&gt;

[root@www ~]# /etc/init.d/httpd restart
```

接下來你只要在你的網址列輸入主機名稱後面加上 http://hostname/server-status 即可發現如下的模樣：

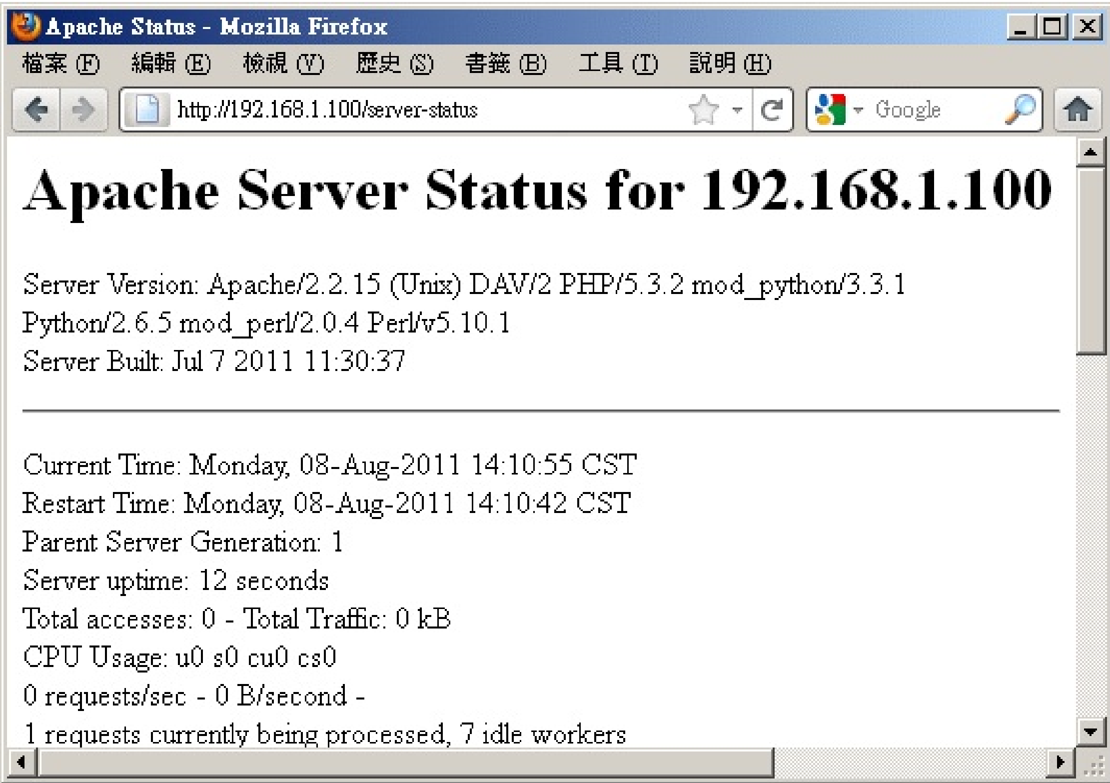


圖 20.3-2、伺服器目前的狀況回報網頁

輸出的結果包括目前的時間以及 Apache 重新啓動的時間，還有目前已經啓動的程序等等，還有網頁最下方會顯示每個程序的用户端與伺服器端的連線狀態。雖然顯示的狀況挺陽春，不過該有的也都有了， 可以讓你約略瞭解一下伺服器的狀況囉。要注意喔，可查閱者 (allow from 的參數) 還是需要限制的比較嚴格一點啦！

20.3.6 .htaccess 與認證網頁設定

對於保護 Apache 本身的資料方面，除了上述的 Order 以及 Limit 之外，還有什麼方式呢？因爲 Order 與 Limit 主要是針對 IP 網域或者是主機名稱來管理，那如果我們用户端是使用撥接方式取得 IP， 那麼 IP 會一直變動的，如此一來那個保護的目錄使用者也就不能在任何地方進入了，會造成一些困擾。

此時如果能夠使用密碼保護的方式，讓使用者可以輸入帳號/密碼即可取得瀏覽的權限的話，那用户端就不用受到那個 order 的 Allow, deny 的限制啦！眞好～呵呵！Apache 確實剛好有提供一個簡單的認證功能， 讓我們可以輕鬆愉快的就設定好密碼保護的網頁呢！

Tips: 什麼是受保護的資料呢？舉例來說，學校老師們可能會提供一些教學教材或者是習題給同學， 這些資料不想給所有人取得，那麼就可以將這些資料放在特定的受保護的目錄中。還有例如某些重要的 Apache 伺服器分析的資料 (本章後面提及的一些分析工具)，這些資料建置的方法需要啓用 CGI 程式， 而 CGI 程式的執行是有風險的，而且那些分析所得的資料也很重要。此時，該程式與輸出結果就需要放在受保護的目錄啦！



那麼那個認證網頁如何搞定？簡單的説，他要這樣處理：

1. 建立受保護的目錄：既然我們是『按了某個連結進入某個目錄之後，才會出現對話視窗』， 那麼首先當然就是要有那個設定爲認證網頁的『目錄』囉！請注意，是要目錄才行喔！
2. 設定 Apache 所需參數：然後，在對話視窗中，既然我們需要輸入帳號與密碼， 那麼自然就需要密碼檔囉！另外，雖然 Apache 有支援 LDAP 及 MySQL 等等的認證機制，不過我們這裡並不討論其他的認證機制， 完全使用 Apache 的預設功能而已，所以，底下我們會使用基本 (Basic) 的認證模式喔！
3. 建立密碼檔案：處理完基本的設定後，再來則是建立登入時所需要的帳號與密碼！
4. 最後，重新啓動 Apache 就 OK 啦！

其中，第二個步驟會比較有趣，我們説過，任何的設定資料都可以直接寫到 httpd.conf 這個設定檔當中， 所以設定保護目錄的參數資料確實可以寫入 httpd.conf 當中。不過，想一想，如果你的 Apache 伺服器有 30 個使用者具有個人首頁，然後他們都需要製作保護目錄，那個 httpd.conf 只有身爲 root 的你才能夠修改， 更可怕的是『每次改完都需要重新啓動 Apache』～請問，你的時間精力是否會受到『很嚴厲的考驗？』

所以啦，如果我們能夠透過外部的檔案來取代設定 httpd.conf 内的參數，那麼是否會比較好？ 而且最好能夠該檔案設定即生效，不需要重新啓動 Apache 的話，那就更好啦！因爲如此一來， 你就可以交給使用者自行管理他們的認證網頁囉！呵呵～透過 httpd.conf 内的

AllowOverride 參數，配合 .htaccess 這個檔案的設定就 OK 搞定！這個設定項目與設定檔 httpd.conf 的關係可以這樣看：

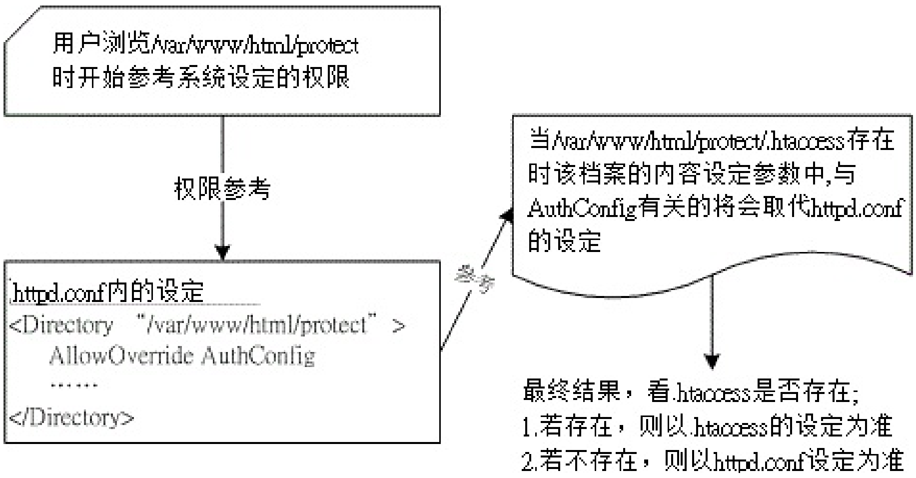


圖 20.3-3、.htaccess 與主要設定檔 httpd.conf 的相關性

也就是說：

- 主設定檔 httpd.conf 的修訂：你必需要在 httpd.conf 這個主設定檔當中先以 AllowOverride 指定某個目錄下的 .htaccess 能夠進行取代的參數爲何？一般有 AuthConfig, Options 等等，考慮到系統資料的安全，建議提供 AuthConfig 的項目就好了。設定完畢後請重新啓動 Apache。
- .htaccess 放置的目錄：在受保護的目錄底下務必要存在 .htaccess 這個檔案，透過這個檔案即可修改 httpd.conf 內的設定啊！
- .htaccess 的修改：.htaccess 設定完『立刻生效』，不需要重新啓動 Apache，因爲該檔案的內容是『當有用户端瀏覽到該目錄時，該檔案才會被使用來取代原有的設定。

既然 .htaccess 的用途比較廣，所以底下我們不介紹 httpd.conf 的認證參數了，請你自行測試即可。底下主要以 .htaccess 檔案的設定爲主喔！趕緊來看看吧！

- 1. 建立保護目錄的資料

假設我要將受保護的資料放置到 /var/www/html/protect 當中，記得，這個目錄要讓 Apache 可以瀏覽到才行。 所以你可以立刻將一些重要的資料給他搬移到這裡來。我們先這樣測試一下吧！建立個簡單的測試網頁即可。

```
[root@www ~]# mkdir /var/www/html/protect
[root@www ~]# vim /var/www/html/protect/index.html
&lt;html&gt;
&lt;head&gt;&lt;title&gt;這是個測試網頁啊！&lt;/title&gt;&lt;/head&gt;
&lt;body&gt;看到這個畫面了嗎？如果看到的話，表示你可以順利進入本受保護網頁啦！
&lt;/body&gt;&lt;/html&gt;
```

- 2.1 以 root 的身份處理 httpd.conf 的設定資料

這個動作僅有 root 能作啦！你要開始編輯 httpd.conf ，讓受保護的那個目錄可以使用 .htaccess 啊！

```
[root@www ~]# vim /etc/httpd/conf/httpd.conf
# 確定底下這幾行是存在的，約在 400 行左右！
AccessFileName .htaccess
&lt;Files ~ "^\.ht"&gt;
    Order allow,deny
    Deny from all
    Satisfy All
&lt;/Files&gt;

# 在某個不受影響的地方加入這一段：
&lt;Directory "/var/www/html/protect"&gt;
    AllowOverride AuthConfig
    Order allow,deny
    Allow from all
&lt;/Directory&gt;

[root@www ~]# /etc/init.d/httpd restart  &lt;==重新啟動，不要忘記了！
```

這樣就設定妥當了，很簡單吧！再接下來要準備 .htaccess 的建立了。

- 2.2 建立保護目錄下的 .htaccess 檔案：只要有權限建立者即可進行

要注意，這個檔案是在保護目錄底下喔！不要放錯地方啦！所以你要這樣做：

```
[root@www ~]# cd /var/www/html/protect
[root@www protect]# vim .htaccess
# 只要加入底下這幾行即可
AuthName     "Protect test by .htaccess"
Authtype     Basic
AuthUserFile /var/www/apache.passwd
require user test
```

這些參數的意義是這樣的：

- AuthName：在要你輸入帳號與密碼的對話視窗中，出現的『提示字元』
- AuthType：認證的類型，我們這裡僅列出 Apache 預設的類型，亦即是『basic』的啦
- AuthUserFile：這個保護目錄所使用的帳號密碼設定檔。 也就是說，這個檔案是隨便你設定的，當然啦，所以使用者當然可以自行設定帳號與密碼囉。 檔案內的帳號不限在 /etc/passwd 出現的使用者！另外，這個檔案不要放置在 Apache 可以瀏覽的目錄內，所以我將他放置在首頁之外！避免被不小心竊取。
- require：後面接可以使用的帳號。假如 /var/www/apache.passwd 內有三個帳號， 分別是 test, test1, test2 ，那我這裡只寫了 test ，因此 test1, test2 將無法登入此目錄。 如果要讓該密碼檔內的使用者都能夠登入，就改成『require valid-user』即可啊！

設定好就立刻生效了，不需要重新啟動任何東西啊！

- 3. 建立密碼檔案 htpasswd (只要有權限即可執行)

Apache 預設讀取的帳號/密碼設定資料是由 htpasswd 所建立的，這個指令的語法是這樣的：

```
[root@www ~]# htpasswd [-cmdD] 密碼檔檔名 使用者帳號
選項與參數：
-c  ：建立後面的密碼檔案。如果該檔案已經存在，則原本的資料會被刪除！
      所以如果只是要新增使用者(檔案已存在時)，不必加上 -c 的參數！
-m  ：不使用預設的 CRYPT 加密，改用 MD5 方式加密密碼！
-d  ：使用更複雜的 SHA 方式來加密！
-D  ：刪除掉後面接的那個使用者帳號！

# 1\. 建立 apache.passwd ，帳號爲 test
[root@www ~]# htpasswd -c /var/www/apache.passwd test
New password:  &lt;==這裡輸入一次密碼，注意，螢幕不會有任何訊息。
Re-type new password:  &lt;==這裡再輸入一次
Adding password for user test

[root@www ~]# cat /var/www/apache.passwd
test:FIquw/..iS4yo   &lt;==你瞧瞧！已經建立一個新使用者！

# 2\. 在已存在的 apache.passwd 內增加 test1 這個帳號：
[root@www ~]# htpasswd /var/www/apache.passwd test1
```

再次強調，這個檔案檔名需要與 .htaccess 內的 AuthUserFile 相同，且不要放在瀏覽器可以瀏覽到的目錄！ 這樣就算設定完畢啦！你可以使用瀏覽器在網址列輸入：『http://your.hostname/protect』試看看，結果會如何？ 會像底下這個樣子：



圖 20.3-4、瀏覽到受保護的目錄時，瀏覽器出現的提示視窗示意圖

如果你曾經瀏覽過這個目錄了，當時可能尚未製作保護的檔案，或者是檔案設計錯誤，導致你曾經可以瀏覽該網頁， 則該網頁會被你的瀏覽器快取 (cache) 起來，所以可登入的畫面會一再地出現而不會跑出需要認證的對話視窗。此時你應該要：

- 務必將全部的瀏覽器都關閉，再重新啓動瀏覽器。因爲你成功的登入該目錄後，該次登入的資訊會快取在這次的連線上喔！
- 可以將瀏覽器上頭的『reload (重新讀取)』按下去，讓瀏覽器重新讀取一次；否則快取不會更新。
- 可以將瀏覽器的快取資料全部清除，關閉瀏覽器後再重新啓動瀏覽器看看。

如果還是一直出問題，那就只好前往登錄檔 (/var/log/httpd/error*log)* 察看錯誤資訊囉。 常見的錯誤只是打錯字啦！@@

20.3.7 虛擬主機的設定 (重要！)

接下來我們要談的是『主機代管』...瞎密？不是啦～是一個稱爲虛擬主機的東西啦～這東西很有用喔！他可以讓你的一部 Apache 看起來像有多個『主站首頁』的感覺啦！

- 什麼是虛擬主機 (Virtual Host)

所謂的虛擬主機，基本上就是『讓你的一部伺服器上面，有好多個 "主網頁" 存在，也就是說，硬體實際上只有一部主機，但是由網站網址上來看，則似乎有多部主機存在的樣子！』。舉個例子來說好了，鳥哥提供的網站主要有主要學習網站以及新手討論區，分別在底下的連結：

- 主網站：http://linux.vbird.org
- 討論區：http://phorum.vbird.org

這兩個連結你給他點下去，會發現其實是不同的資料內容，不過，如果你用 dig 之類的軟體來查驗 IP 的話，會發現這兩個網址都指向同一個 IP ㄟ！怎麼會這樣？沒錯啊！這就是虛擬主機的主要功能！他可以讓你的多個主機名稱對應到不同的主網頁目錄 (DocumentRoot 參數)，所以看起來會像有多部實際主機的模樣啦！這樣說，瞭解虛擬主機了嗎？

- 架設的大前提：同一個 IP 有多個主機名稱啦！

那麼要架設虛擬主機需要什麼咚咚呢？以剛剛鳥哥的網站的結果爲例，我必需要有多個主機名稱對應到同一個 IP 去， 所以說，你必需先擁有多個主機名稱才行。要如何擁有多個主機名稱？那就是：

- 向 ISP 申請多個合法的主機名稱，而不自己架設 DNS；
- 自行設定經過合法授權的 DNS 主機來設定自己所需要的主機名稱。

相關的 DNS 申請與設定技巧我們在前幾章都談過了， 你可得自行去瞧瞧先！

- 一個架設範例練習：

我們在第十九章 DNS 裡面不是有設定了多個主機名稱嗎？ 那些主機名稱就是爲了要在這裡實作用的啦！ ^_^！你得要注意的是，我的每個主機名稱都必需要對應到某個主網頁目錄，底下則是鳥哥的一個簡單範例：

主機名稱	對應的主目錄
linux.centos.vbird	/var/www/html
www.centos.vbird	/var/www/www
ftp.centos.vbird	/var/ftp (較特殊)

接下來就是開始設定囉！要告訴你的是，建議你將虛擬主機的設定建立一個新的檔案在 /etc/httpd/conf.d/*.conf* 當中，因爲如此一來你的虛擬主機設定檔就可以進行搬移， 修改的時候也不會影響到原有的 *httpd.conf* 的資料！而因爲 *httpd.conf* 內有個 *Include* 的參數將 */etc/httpd/conf.d/*.conf 的檔案都讀入設定檔當中，所以設定上面就變的很輕便， 備份與升級的時候也比較容易處理嘛！不囉唆，趕緊來實驗一下先！

```
# 1\. 先建立所需要的目錄：
[root@www ~]# mkdir /var/www/www &lt;==www.centos.vbird 所需！
[root@www ~]# yum install vsftpd &lt;==/var/ftp 可由系統軟體提供
[root@www ~]# echo "www.centos.vbird" &gt; /var/www/www/index.html
[root@www ~]# echo "ftp.centos.vbird" &gt; /var/ftp/index.html
# 原有的首頁 (/var/www/html) 就不更動了！另建兩個不同的首頁內容，可供測試用。

# 2\. 開始編輯設定檔，這裡鳥哥用額外的檔案來設定喔！
[root@www ~]# vim /etc/httpd/conf.d/virtual.conf
# 底下這一行在規定『本機任何介面的 port 80 所指定的虛擬主機』的意思。
NameVirtualHost *:80

# 先針對兩個多出來的可瀏覽目錄進行權限方面的規範啊！
&lt;Directory "/var/www/www"&gt;
    Options FollowSymLinks
    AllowOverride None
    Order allow,deny
    Allow from all
&lt;/Directory&gt;
&lt;Directory "/var/ftp"&gt;
    Options FollowSymLinks Indexes
    AllowOverride None
    Order allow,deny
    Allow from all
&lt;/Directory&gt;

# 針對三部主機的 DocumentRoot 進行定！
&lt;VirtualHost *:80&gt;
    ServerName    linux.centos.vbird
    DocumentRoot  /var/www/html
&lt;/VirtualHost&gt;
&lt;VirtualHost *:80&gt;
    ServerName    www.centos.vbird
    DocumentRoot  /var/www/www
    CustomLog     /var/log/httpd/www.access_log combined
    # 不同的主頁可以指定不同的登錄檔資訊，這樣比較好 debug 與分析啦！
&lt;/VirtualHost&gt;
&lt;VirtualHost *:80&gt;
    ServerName    ftp.centos.vbird
    DocumentRoot  /var/ftp
&lt;/VirtualHost&gt;

[root@www ~]# /etc/init.d/httpd restart
```

你要注意的只有幾點：

1. 在虛擬主機的設定上還有很多的可用的功能，不過，最低的限度是需要有 ServerName 及 DocumentRoot 這兩個即可！
2. 使用了虛擬主機後，原本的主機名稱 (linux.centos.vbird) 也要同時寫入虛擬主機的對應中， 否則這個主機名稱可能會不知道被丟到哪裡去喔！
3. 在 www.centos.vbird 這個主機當中多了個 CustomLog ，表示任何向 www.centos.vbird 要求資料的記錄都會改寫入 /var/log/httpd/www.access_log 而不是預設的 /var/log/httpd/access_log。 但這個新增的登錄檔必需要加入 logrotate 的管理當中才行喔！否則登錄檔會大到『爆表』

接下來，只要你用户端的瀏覽器可以找到這三個主機名稱並連線到正確的 IP 去， 你這個 Apache 就可以同時提供三個網站的站址了，很方便吧！^_^。

- 虛擬主機常見用途

虛擬主機爲什麼會這麼熱門啊？這是因爲他可以進行底下的任務：

- 主機代管： 如果你有一部很快速的電腦，配合你的網路頻寬又大的話，那麼你可以用這個虛擬主機的技術來『拉客』喔！ 因爲畢竟不是所有公司都有維護伺服器的能力，如果你能夠提供合理的流量、親和的資料傳輸介面、 穩定的提供服務，並且給予類似 MySQL 資料庫的支援，那麼當然有可能進行『主機代管』的業務啊！你說是吧！ ^_^
- 伺服器資料備援系統： 你可以在兩個地方放置兩部主機，主機內的網頁資料是一模一樣的 (這個可以使用 rsync 來達成的) ，那麼你將可以利用 Apache 的虛擬主機功能， 配合 DNS 的 IP 指向設定，讓某一部主機掛點時，另外一部主機立刻接管 WWW 的要求！ 讓你的 WWW 伺服器不會有任何斷線的危機啊！ ^_^ (註：當 A 伺服器掛點時， 趕緊設定 DNS ，讓原本 A 的 IP 指定給 B，則任何向該 IP 要求的 WWW 將會被導向 B， B 有 A 的備份資料以及虛擬主機設定，搞定！)
- 將自己的資料分門別類： 如果野心沒有這麼大的話，那麼如果你有幾個不同的資料類型時，也可以利用虛擬主機將各種資料分門別類啦！ 例如將部落格指向 blog.centos.vbird，將討論區指向 forum.centos.vbird ，將教學資料指向 teach.centos.vbird 等等，這樣的網址就很容易讓用户端瞭解啦！你說是吧！ ^_^

20.4 登錄檔分析以及 PHP 強化模組

除了這些基本的 Apache 使用方式之外，我們還有哪些事情可以玩的？當然還有很多啦！包括有趣的 PHP 效能強化模組、登錄檔分析以瞭解整個 Apache 的使用情況等等！讓我們來瞧一瞧！

20.4.1 PHP 強化模組 (eaccelerator) 與 Apache 簡易效能測試

雖然 PHP 網頁程式標榜的是速度快速，不過因爲 PHP 畢竟是先將一些可用函數先編譯成爲模組，然後當網頁使用到該 PHP 程式的時候，再由呼叫 PHP 模組來達成程式所需要的行爲。由於多了一道手續， 所以他的執行效能還是有別於傳統編譯的程式語言囉。

那麼如果我們可以將 PHP 程式預先轉換成爲可直接執行的 binary file，不就可以直接讀取進而加快速度嗎？ 沒錯！是這樣～這東西稱爲預編器～其中有一套軟體稱爲 eaccelerator，eaccelerator 可以將你的 PHP 程式與 PHP 核心及相關函式庫預先編譯後暫存下來，以提供未來使用時可以直接執行，加上他可以優化你的 PHP 程式，因此，可以讓你的 PHP 網頁速度增快不少喔！eaccelerator 的官方網站在底下：

- http://eaccelerator.net/

整個安裝的流程很簡單啦！你先將這個軟體的原始碼下載下來，我這裡假設你將他下載到 /root 目錄下， 另外你必需要確定你有安裝 php-devel, autoconf, automake, m4, libtool 等軟體才行！那就趕緊來安裝吧！(鳥哥是以 0.9.6.1 這一版爲範例的喔！)

```
# 1\. 解壓縮檔案，並且進行 patch 的動作：
[root@www ~]# cd /usr/local/src
[root@www src]# tar -jxvf /root/eaccelerator-0.9.6.1.tar.bz2
[root@www src]# cd eaccelerator-0.9.6.1/

# 2\. 利用 phpize 進行 PHP 程式的預處理
[root@www eaccelerator-0.9.6.1]# phpize
# 過程會出現一些警告資訊，不要理他沒關係！
[root@www eaccelerator-0.9.6.1]# ./configure --enable-eaccelerator=shared \
&gt; --with-php-config=/usr/bin/php-config
[root@www eaccelerator-0.9.6.1]# make

# 3\. 將他整個安裝起來！
[root@www eaccelerator-0.9.6.1]# make install
# 此時這個新編譯的模組會被放置到 /usr/lib64/php/modules/eaccelerator.so 當中！
```

將模組處理完畢之後接下來就是要讓 PHP 使用這個模組啦！如何進行呢？

```
# 1\. 預先載入這個 PHP 的模組：
[root@www ~]# echo "/usr/lib64/php/modules/" &gt;&gt; \
&gt; /etc/ld.so.conf.d/php.conf

[root@www ~]# ldconfig
# 關於 [ld.so.conf 以及 ldconfig](http://linux.vbird.org/linux_basic/0520source_code_and_ta

# 2\. 修改 php.ini 喔！
[root@www ~]# vim /etc/php.ini
# 在這個檔案的最底下加入這幾行：
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; http://eaccelerator.net/       ;
; 2011/08/08 VBird               ;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
extension="eaccelerator.so"
eaccelerator.shm_size="16"
eaccelerator.cache_dir="/tmp/eaccelerator"
eaccelerator.enable="1"
eaccelerator.optimizer="1"
eaccelerator.check_mtime="1"
eaccelerator.debug="0"
eaccelerator.filter=""
eaccelerator.shm_max="0"
eaccelerator.shm_ttl="0"
eaccelerator.shm_prune_period="0"
eaccelerator.shm_only="0"
eaccelerator.compress="1"
eaccelerator.compress_level="9"

# 3\. 建立 eaccelerator 的暫存資料，重點在於權限要設定正確！
[root@www ~]# mkdir /tmp/eaccelerator
[root@www ~]# chmod 777 /tmp/eaccelerator
[root@www ~]# /etc/init.d/httpd restart
```

基本上這樣就設定妥當啦！要注意的是：『因爲你的 eaccelerator 是根據目前這一版的 PHP 核心所編譯出來的，所以未來如果你的 Linux distribution 有釋出新版的 PHP 時，你也順利更新到新版的 PHP 了，那你的這個 eaccelerator 就必需要自行手動再更新一次， 以配合到正確的 PHP 版本，否則這個模組將不會正確運作。』！很重要喔！

那如何確認這個模組有正確的在運作呢？你可以利用 20.2.4 小節談到的 phpinfo() 這個函式來查閱，透過瀏覽器你應該會看到如下的畫面：

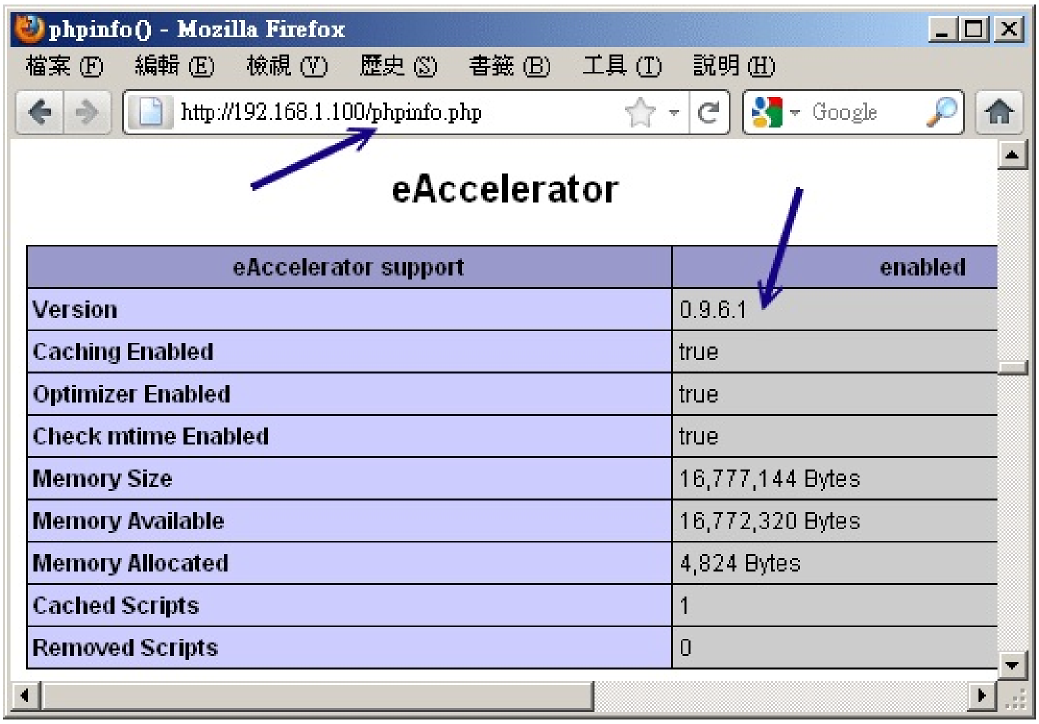


圖 20.4-1、確定 eaccelerator 有運作的畫面

如果你的 eaccelerator 沒有啟動的話，那就看不到上圖的畫面啦！藉由這個動作來測試測試吧！ ^_^！ 接下來我們利用 Apache 提供的一個小程式來測試一下我們網站的效能吧！這個程式叫做 ab ， 他可以主動的向主機重複要求多筆資料來確認主機的效能喔！

```
[root@www ~]# ab [-dSk] [-c number] [-n number] 網頁檔名
選項與參數：
-d ：不要顯示 saved table 的百分比資料；通常不要那個資料，所以會加 -d
-k ：還記得上面的 KeepAlive 吧！加入 -k 才會以這樣的功能測試；
-S ：不顯示長訊息，僅顯示類似 min/avg/max 的簡短易懂訊息！
-c ：同時有多少個『同時連線』的設定(可想成同時連線的 IP )
-n ：同一個連線建立幾個要求通道！(可想成同一個 IP 要求的幾條連線)
更多的訊息請自行 man ab 喔！

# 針對我們剛剛測試時的 phpinfo.php 這個檔案來測試！
[root@www ~]# ab -dSk -c100 -n100 http://localhost/phpinfo.php
This is ApacheBench, Version 2.3 &lt;$Revision: 655654 $&gt;
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/
....中間省略....
Document Path:          /phpinfo.php
Document Length:        54204 bytes
....中間省略....
Total transferred:      5436100 bytes
HTML transferred:       5420400 bytes
Requests per second:    39.97 [#/sec] (mean)
Time per request:       2501.731 [ms] (mean)
Time per request:       25.017 [ms] (mean, across all concurrent requests)
Transfer rate:          2122.01 [Kbytes/sec] received
....底下省略....
```

根據這個軟體的輸出你會知道每秒鐘的傳輸速率、最大傳輸速度等等，可以約略知道一下基本效能啦！ 不過鳥哥這個程式是在自己機器上面測試的，速度快是正常的！你可以在網路的另一頭來測試一下說！ (註：這個 ab 程式對於讀取 MySQL 之類的網頁似乎沒有辦法成功的完成測試的樣子， 你應該以較單純的網頁來測試吧！)

20.4.2 syslog 與 logrotate

請特別注意，我們的 Apache 登錄檔主要記錄兩個東西，分別是：

- /var/log/httpd/access_log ：用户端正常要求的記錄資訊
- /var/log/httpd/error_log ：使用者錯誤要求的資料，包括伺服器設定錯誤的資訊等。

那個 /var/log/httpd/error*log* 可以讓你處理很多設定錯誤的情況，包括網頁找不到、 檔案權限設定錯誤、密碼檔案檔名填錯等等。至於 *access_log* 則可以讓你分析那個網頁最熱門！ ^^！不過你可得注意的是：『在稍有規模的網站下，Apache 的登錄檔每週記錄量甚至可達 1GB 以上』的紀錄。以鳥哥的主網站來說，一個星期逼近 1GB 的登錄檔是合理的...

不過，因爲登錄檔是純文字資訊，所以如果能夠給予壓縮的話，那麼備份下來的登錄檔將可以減少到數十MB而已， 這樣可大大的減少了磁碟空間的浪費啊！如果你是使用預設的 Apache 來處理你的伺服器時， 那麼系統已經作了一個 logrotate 給你使用了，如果你是使用 Tarball 自己安裝的， 那麼...你就得要自行手動建立底下這個檔案啦！鳥哥底下是以 CentOS 6.x 提供的檔案來作說明的：

```
[root@www ~]# vim /etc/logrotate.d/httpd
/var/log/httpd/*log {
    missingok
    notifempty
    compress    &lt;==建議加上這一段，讓你的備份登錄檔可以被壓縮
    sharedscripts
    delaycompress
    postrotate
        /sbin/service httpd reload &gt; /dev/null 2&gt;/dev/null &#124;&#124; true
    endscript
}
```

爲什麼這裡很重要呢？鳥哥的伺服器曾經發生過一件事情，就是....突然 WWW 效能變很差！後來追蹤的原因竟然是... /var/ 的容量被用完了！而耗掉這個 partition 的元兇竟然是 Apache 的登錄檔！ 當時 /var/ 僅給 5GB ，而每個星期的登錄檔就上達 1GB 以上，備份四個星期的結果， /var/ 想不爆掉也很難～ 所以啦，建議你的 /var 要給個 10GB 以上才好吶！而且備份登錄檔也要壓縮才好吶！

Tips: 關於 syslog 與 logrotate 的詳細說明請參考基礎篇的內容喔！或者是到底下的連結：
http://linux.vbird.org/linux_basic/0570syslog.php



此外，透過分析登錄檔其實我們可以知道我們的網站到底是哪一個網頁最熱門？也且也能知道用戶端是來自哪裡呢！目前針對 Apache 有很多的分析軟體，我們底下僅介紹兩個常見的分析軟體給大家吻！

20.4.3 登錄檔分析軟體：webalizer

事實上，CentOS 6.x 預設就提供了 webalizer 這個分析軟體了！你只要將這套軟體安裝上來就是了。 如果你不是使用 CentOS 呢？沒關係，官方網站上也可以下載，安裝也很簡單！

- 官方網站：http://www.mrunix.net/webalizer/
- 設定難度：簡單，極適合新手架設
- 軟體特色：大致上，所有分析的內容他都有了！雖然圖表比較沒有那麼炫...
- 授權模式：GPL

CentOS 6.x 提供的這個軟體設定檔在 /etc/webalizer.conf ，而且他設定每天會分析一次 WWW 的登錄檔， 不過這個軟體預設會將輸出的結果放置到 /var/www/usage ，並且這個目錄僅有本機可以查閱， 鳥哥並不喜歡這樣的設定。我們剛剛不是有建立一個保護目錄 /var/www/html/protect 嗎？ 這個目錄的功能來啦！鳥哥預計將 webalizer 的輸出資料放置到 /var/www/html/protect/webalizer 底下去， 所以知道密碼的都能夠查閱呢！整個動作是這樣的：

```
# 1\. 先處理設定檔，變更指定一下我們要輸出的目錄即可：
[root@www ~]# vim /etc/webalizer.conf
# 確定一下底下這幾行是正確的！其他的則保留預設值
LogFile        /var/log/httpd/access_log        &lt;==約在 28 行
OutputDir      /var/www/html/protect/webalizer  &lt;==約在 42 行
Incremental    yes                              &lt;==約在 67 行

# 2\. 建立該保護目錄的資料：
[root@www ~]# cp -a /var/www/usage/ /var/www/html/protect/webalizer
[root@www ~]# /etc/init.d/httpd restart

# 3\. 開始測試執行 webalizer 的分析工作
[root@www ~]# webalizer
```

現在請你在瀏覽器上面輸入： http://your.hostname/protect/webalizer ，看看輸出的結果是如何吧！ 結果應該會如下所示：

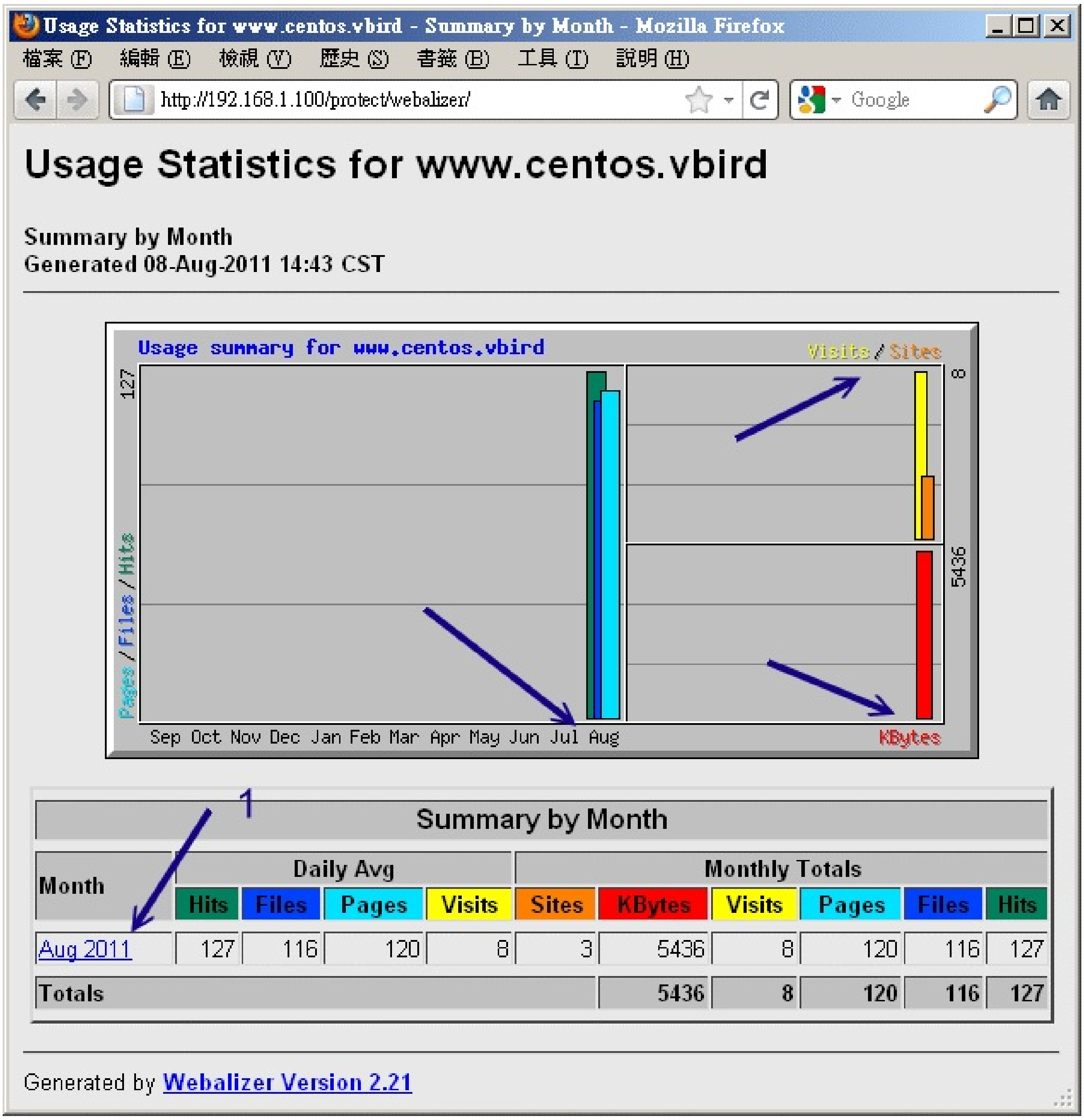


圖 20.4-2、webalizer 分析工具所得的分析畫面

在上圖當中的箭頭 1 處你還可以點選喔，點選後會告知你當月的各項分析結果，很不錯吧！

20.4.4 登錄檔分析軟體：awstats

除了 webalizer 之外，我們其實還可以透過 awstats 這個厲害到不行的 perl 的程式來進行資料分析， 由於這個軟體是以 perl 來執行的，所以請確定你的 mod_perl 已經安裝且 CGI 的執行權限已經啓動了！ 這個軟體的特色是：

- 官方網站：http://awstats.sourceforge.net/
- 官方軟體：http://awstats.sourceforge.net/#DOWNLOAD
- 設定難度：較難，需要有點技巧！
- 軟體特色：中文化的很完整，而且該有的都有了，相當炫的一個分析利器！

- 授權模式：GPL

這套軟體不但可以由系統的 cron 來進行分析，甚至還提供瀏覽器直接以 CGI 的方式來即時更新登錄檔吶！ 眞是厲害厲害！鳥哥個人是比較不喜歡使用瀏覽器來線上更新分析的結果， 因爲在你更新分析結果時，怎麼知道系統會不會很忙碌？如果系統正在忙碌中， 這套軟體的分析可也是很耗費系統資源的吶！所以建議直接以 crontab 的方式來處理即可。

目前官方網站不但提供 tarball 甚至也提供 RPM 來給使用者下載了！眞是方便啊！ 但是你還是要注意的，這個軟體曾經因爲安全性的問題導致很多網站的掛點， 所以建議你還是把這個軟體的輸出結果放置在受保護的目錄中喔！底下鳥哥以 7.0-1 這個 RPM 版本來說明， 請你自行到官方網站下載吧！(註：檔名爲 awstats-7.0-1.noarch.rpm)

假設你將這個 RPM 檔案放置到 /root 當中，那麼自己 rpm -ivh filename 去安裝他吧！ 不要跟我說你不會 RPM ～鳥哥是會昏倒的～@_@！由於這個 RPM 檔案將 awstats 的資料通通放置到 /usr/local/awstats 當中去了！ 爲了自己網頁設定上的方便，建議你是可以這樣做的：

```
# 1\. 先安裝後再將 awstats 提供的 Apache 設定資料給他複製到 conf.d 下
[root@www ~]# rpm -ivh awstats-7.0-1.noarch.rpm
[root@www ~]# cp /usr/local/awstats/tools/httpd_conf  \
&gt; /etc/httpd/conf.d/awstats.conf
[root@www ~]# vim /etc/httpd/conf.d/awstats.conf
Alias /awstatsclasses "/usr/local/awstats/wwwroot/classes/"
Alias /awstatscss "/usr/local/awstats/wwwroot/css/"
Alias /awstatsicons "/usr/local/awstats/wwwroot/icon/"
Alias /awstats/ "/usr/local/awstats/wwwroot/cgi-bin/"
&lt;Directory "/usr/local/awstats/wwwroot"&gt;
    Options +ExecCGI
    AllowOverride AuthConfig  &lt;==這裡改成這樣，因爲要保護！
    Order allow,deny
    Allow from all
&lt;/Directory&gt;
[root@www ~]# /etc/init.d/httpd restart
```

awstats 還眞的挺貼心的，因爲他釋出的文件當中就有關於 Apache 的設定資料，我們直接將他放到 conf.d/ 那個目錄下並且更名後，重新啓動 Apache 就生效了！眞方便。再來則是要針對我們的 WWW 登錄檔來設定啦！設定檔其實是在 /etc/awstats 目錄下，在該目錄下有個範例檔爲 awstats.model.conf，其實這個設定檔『檔名』格式爲：

- awstats.主機名稱.conf

因爲鳥哥這部主機名稱爲 www.centos.vbird，所以假設主機名稱爲 www，所以檔名就應該是 awstats.www.conf 囉！請你將他複製一個新檔，然後這樣做：

```
[root@www ~]# cd /etc/awstats
[root@www awstats]# cp awstats.model.conf awstats.www.conf
[root@www awstats]# vim awstats.www.conf
# 找到底下這幾行，並且修改一下內容啊：
LogFile="/var/log/httpd/access_log"  &lt;== 51行：確定登錄檔所在的位置
LogType=W                            &lt;== 63行：針對 WWW 的登錄檔分析
LogFormat=1                          &lt;==122行：Apache 的登錄檔格式
SiteDomain="www.centos.vbird"        &lt;==153行：主機的 hostname
HostAliases="localhost 127.0.0.1 REGEX[centos\.vbird$]"
DirCgi="/awstats"                    &lt;==212行：能夠執行 awstats 的目錄
DirIcons="/awstatsicons"             &lt;==222行：awstats 一些小圖示的目錄
AllowToUpdateStatsFromBrowser=0      &lt;==239行：不要利用瀏覽器來更新！
Lang="tw"                            &lt;==905行：重要！這是語系！
```

接著開始測試一下是否可以產生正確的分析資料出來？

```
[root@www awstats]# cd /usr/local/awstats/wwwroot/cgi-bin
[root@www cgi-bin]# perl awstats.pl -config=www -update  \
&gt; -output &gt; index.html
# 那個 -config 後面接的就是 awstats.www.conf 的意思！會產生 index.html

[root@www cgi-bin]# ls -l
awstats082011.www.txt   &lt;==剛剛才建立的重要資料檔！
awstats.pl              &lt;==就是剛剛我們下達的執行檔！
index.html              &lt;==重要輸出首頁檔案
```

接下來讓我們趕緊來建立保護目錄的 .htaccess 檔案吧！請注意，鳥哥這裡假設你已經有密碼檔了， 所以直接建立檔案即可啊！

```
[root@www ~]# cd /usr/local/awstats/wwwroot
[root@www wwwroot]# vi .htaccess
AuthName     "Protect awstats data"
Authtype     Basic
AuthUserFile /var/www/apache.passwd
require      valid-user
```

之後，只要你輸入『http://your.IP/awstats/』，就能夠看到輸出的圖表了！圖表有點像這樣：

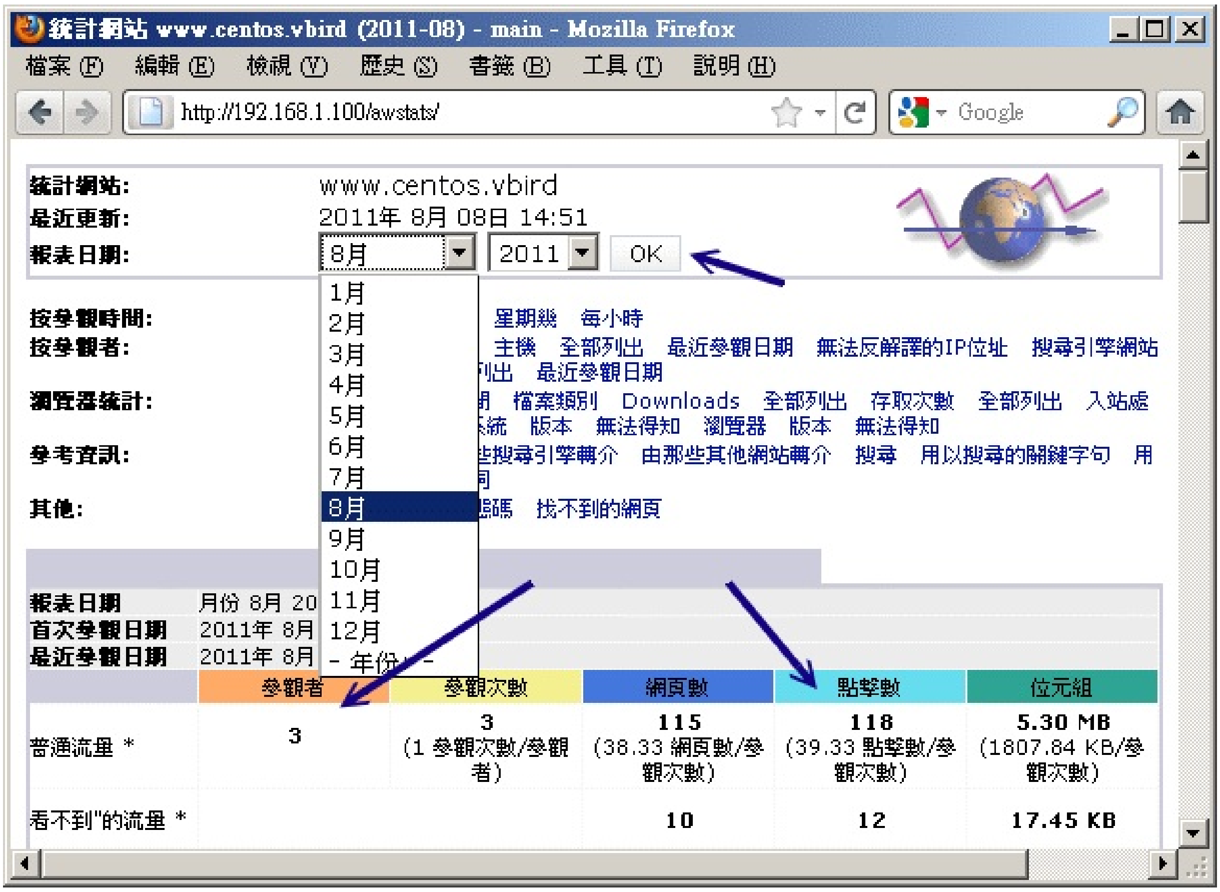


圖 20.4-3、awstats 分析工具所得的分析畫面

事實上，資料非常的多，你可以自行查閱輸出的結果。在上圖當中的箭頭處，你還可以自己選擇曾有的月份資料來進行顯示！ 最後，將分析的動作規定在每天三點的時候跑，你可以這樣做：

```
[root@www ~]# vim /usr/local/awstats/wwwroot/cgi-bin/awstats.sh
cd /usr/local/awstats/wwwroot/cgi-bin
perl awstats.pl -config=www -update -output &gt; index.html

[root@www ~]# chmod 755 /usr/local/awstats/wwwroot/cgi-bin/awstats.sh
[root@www ~]# vim /etc/crontab
0 3 * * * root /usr/local/awstats/wwwroot/cgi-bin/awstats.sh
```

這樣你就知道你的主機到底有多受歡迎囉！ ^_^！另外，再次千萬拜託！這個軟體所在的目錄務必要製作密碼保護！不要隨意釋放出來！ 甚至上面提供的一些目錄的連結你都可以根據自己的主機與喜好來重新修改，會比較安全的啦！

20.5 建立連線加密網站 (https) 及防砍站腳本

從本章一開始的 20.1 就談過 http 這個通訊協定是明碼傳送資料，而那個 https 才是加密傳輸的！那加密的方法是透過 SSL 啊，這個 SSL 就是以 openssl 軟體來提供的一個加密函式庫。更多與 https 有關的資訊，請參考 20.1.4 吧！

20.5.1 SSL 所需軟體與憑證檔案及預設的 https

要達成讓 apache 支援 https 協定的話，你必須要有 mod_ssl 這個軟體才行！請先自行使用 yum 去裝好這個軟體吧！ 並且重新啟動 httpd 喔！同時，我們的 CentOS 6.x 也已經預設提供了 SSL 機制所需要的私鑰與憑證檔案囉！相關軟體提供的檔案如下：

- /etc/httpd/conf.d/ssl.conf：mode_ssl 提供的 Apache 設定檔；
- /etc/pki/tls/private/localhost.key：系統私鑰檔，可以用來製作憑證的！
- /etc/pki/tls/certs/localhost.crt：就是加密過的憑證檔！(signed certificate)

既然系統都已經幫我們搞定了，那麼就讓我們直接來瀏覽一下，看看系統預設提供的 https 是長的什麼模樣吧！ 打開你的瀏覽器，輸入 https://你的IP 來連線看看：

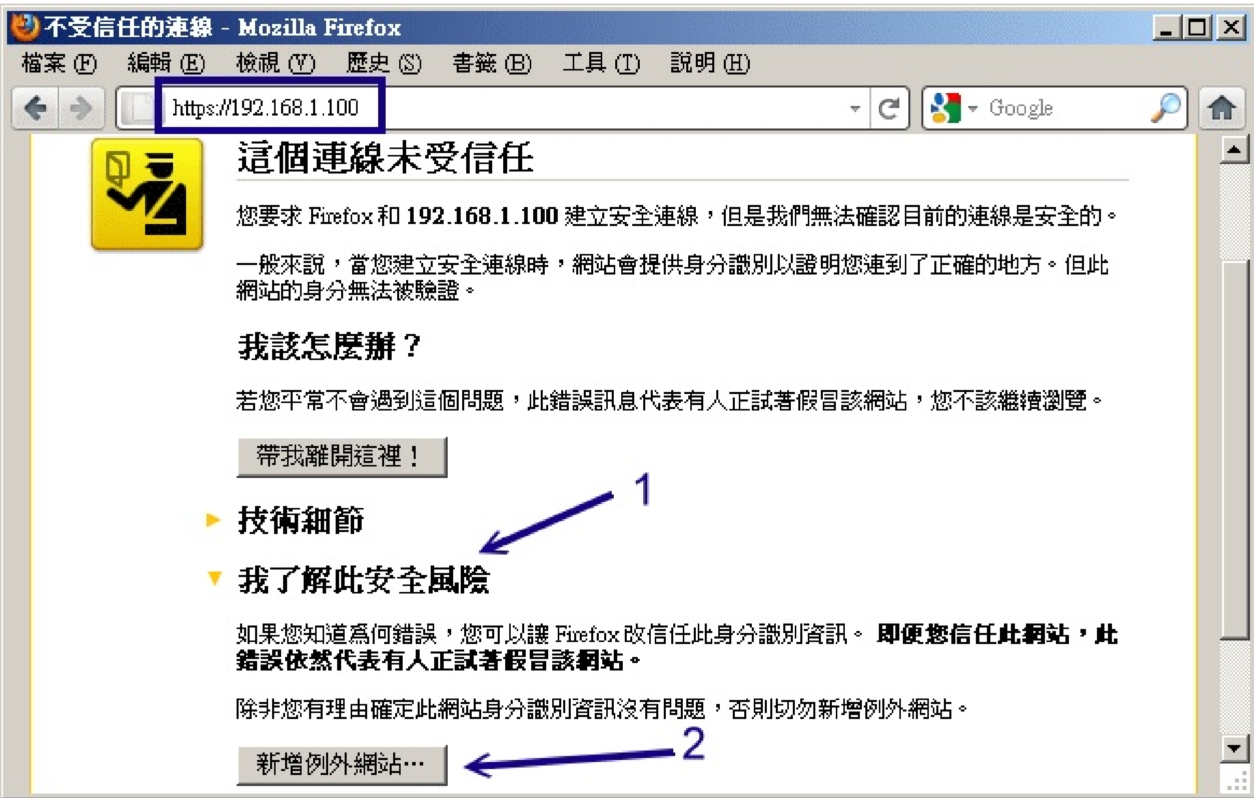


圖 20.5-1、在 firefox 底下看到的 SSL 安全問題圖示

就如同本章 20.1.4 談到的，因爲我們這個 Apache 網站並沒有將此憑證向 CA 註冊，因此就會出現上述的訊息了！ 這就類似 ssh 連線時，系統需要你輸入『 yes 』是一樣的啦！要接受憑證後才能夠進行加密的功能。所以，請點選上圖中的箭頭 1，此時就會延伸出箭頭 2 的位置，按下去吧！然後就會出現如下所示：

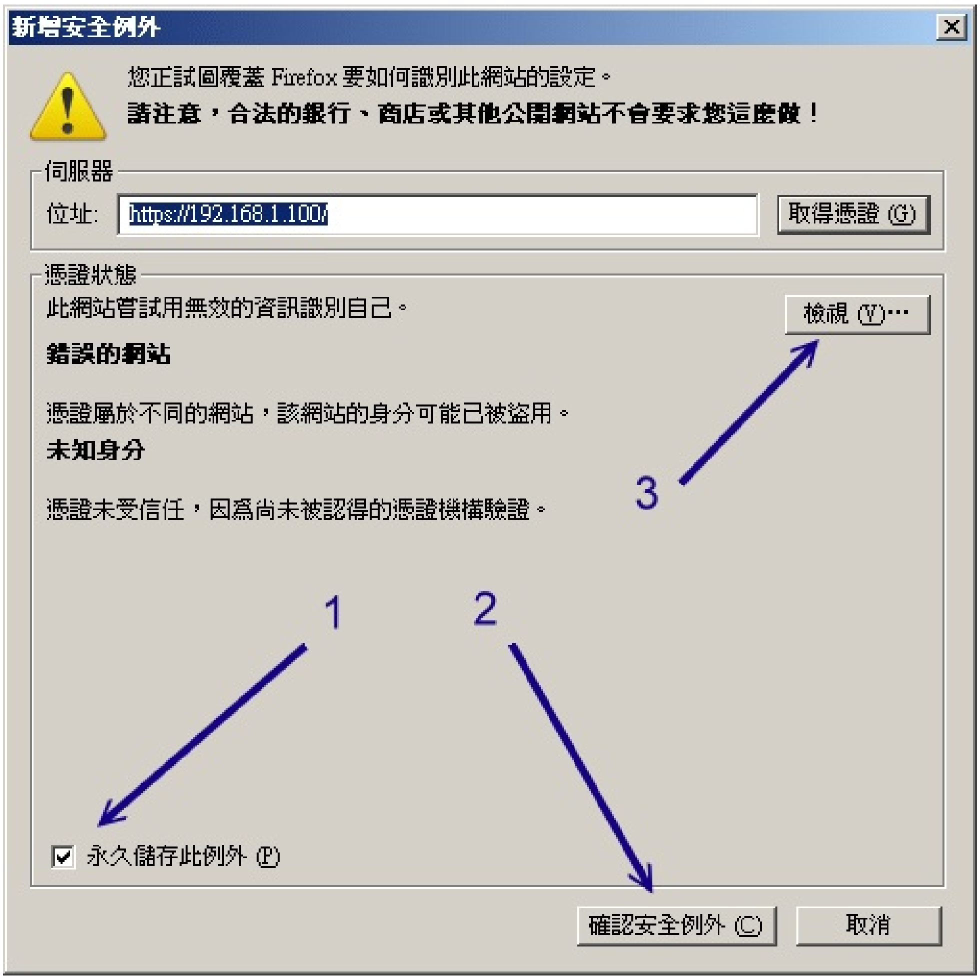


圖 20.5-2、在 firefox 底下接受一把私有的憑證所需要的流程

如果你確定這個網站是你自己的可信任網站，那就按下 1 及 2 的箭頭處！如果還想要看一下這個網站所提供的相關憑證內容， 就按下 3 箭頭的地方：

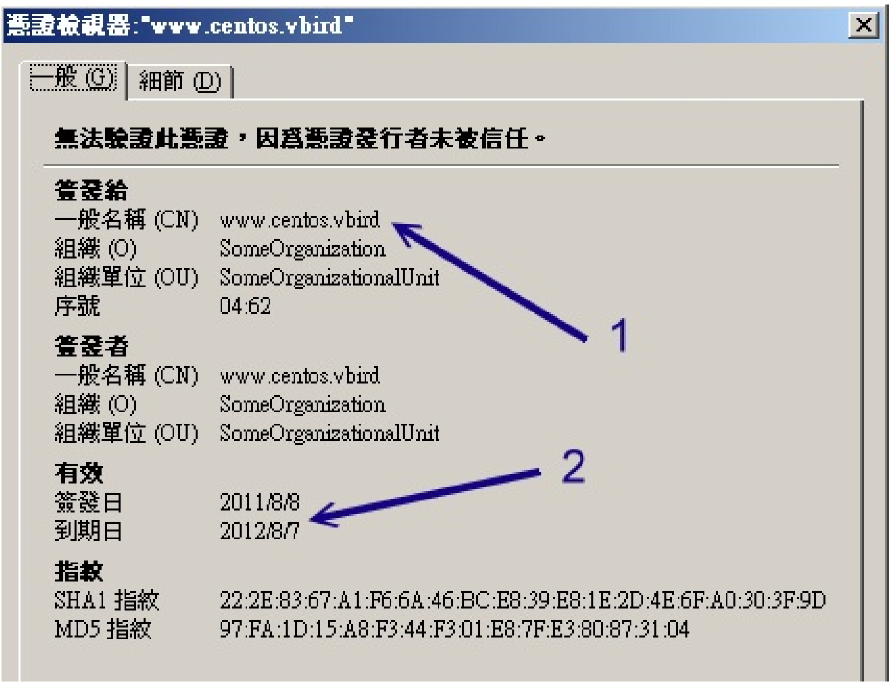


圖 20.5-3、在 firefox 底下觀察憑證的詳細內容

由於這個憑證檔案的建置是在第一次啓動 Linux 時就安裝好了憑證檔，而在 CentOS 6.x 底下，預設的憑證有效期限爲 1 年，所以你就會看到上圖中箭頭 2 所指的，簽發日到到期日共有一年啊！當你按下關閉後，就能夠看到實際的 https:// 提供的網站內容囉！這就是預設的 SSL 網站啦！你的重要資訊可以放在這裡～讓資料在網路上傳輸更佳的安全！

20.5.2 擁有自製憑證的 https

- 建立憑證檔

預設的憑證雖然已經可以讓你順利的使用 https 了，不過，憑證的有效日僅有 1 年而已～實在討厭～ 所以，我們還是得要自製憑證才行～這個憑證的製作僅是私有 WWW 網站的用途，並沒有要拿去 CA 註冊喔！ 那麼自製憑證需要什麼步驟呢？基本上需要的流程是：

1. 先建立一把 private key 預備提供給 SSL 憑證簽章要求所用；
2. 最後建立 SSL 憑證 (test certificates)。

那麼建立憑證有沒有很困難呢？沒有啦！因爲 CentOS 6.x 已經幫我們寫好了 Makefile 了！ 你先到 /etc/pki/tls/certs 這個目錄下，然後直接輸入 make 這個指令，就能夠看到所有可行的目標動作！我們就可以很快速的建置好憑證喔！ 不過，因爲預設的私鑰檔需要加上密碼才能夠進行建立，所以我們還得要額外進行一下動作就是了。好！ 現在假設我們要建立的是名爲 vbird 的憑證！那麼底下流程中，所有的關鍵字就是 vbird！簡單流程如下所示：

```
# 1\. 先到 /etc/pki/tls/certs 去建立一把給 Apache 使用的私鑰檔案：
[root@www ~]# cd /etc/pki/tls/certs
[root@www certs]# make vbird.key
umask 77 ; /usr/bin/openssl genrsa -aes128 2048 &gt; vbird.key  &lt;==其實是這個指令
Generating RSA private key, 2048 bit long modulus
.................................................................+++
...............................+++
e is 65537 (0x10001)
Enter pass phrase:  &lt;==這裡輸入這把私鑰的密碼，需要多於四個字元！
Verifying - Enter pass phrase:  &lt;==再一次！

# 2\. 將剛剛建立的檔案中，裡面的密碼取消掉！不要有密碼存在啦！
[root@www certs]# mv vbird.key vbird.key.raw
[root@www certs]# openssl rsa -in vbird.key.raw -out vbird.key
Enter pass phrase for vbird.key.raw: &lt;==輸入剛剛的密碼啦！
writing RSA key
[root@www certs]# rm -f vbird.key.raw  &lt;==舊的金鑰檔移除
[root@www certs]# chmod 400 vbird.key  &lt;==權限一定是 400 才行！

# 3\. 建置所需要的最終憑證檔！
[root@www certs]# make vbird.crt SERIAL=2011080801
umask 77 ; /usr/bin/openssl req -utf8 -new -key vbird.key -x509 -days 365
-out vbird.crt -set_serial 2011080801  &lt;==可以加入日期序號
You are about to be asked to enter information that will be incorporated
into your certificate request.
-----
Country Name (2 letter code) [XX]:TW
State or Province Name (full name) []:Taiwan
Locality Name (eg, city) [Default City]:Tainan
Organization Name (eg, company) [Default Company Ltd]:KSU
Organizational Unit Name (eg, section) []:DIC
Common Name (eg, your name or your server's hostname) []:www.centos.vbird
Email Address []:vbird@www.centos.vbird

[root@www certs]# ll vbird*
-rw-------. 1 root root 1419 2011-08-08 15:24 vbird.crt  &lt;==最終憑證檔！
-r--------. 1 root root 1679 2011-08-08 15:22 vbird.key  &lt;==系統私鑰檔
```

這樣就建立好憑證檔了！接下來就是得要去處理 ssl.conf 這個設定內容喔！另外，這把憑證依舊只能使用 1 年！如果你想要建立十年的憑證，那就得要修改一下 Makefile 裡面的內容，將 365 改成 3650 即可！

Tips: 如果你曾經多次重複進行上述的建立憑證動作，會發現到同一個憑證內容若製作多次，則最終用户端瀏覽器會出現一些錯誤訊息， 導致無法連線！因此，建議多加一個序號 (SERIAL) 的參數，可以修訂這個錯誤喔！



- 修改 ssl.conf 的內容，使用自製憑證

修改 ssl.conf 的內容也是很簡單！只要修改兩個地方，亦即是檔案檔名的地方即可！

```
[root@www ~]# vim /etc/httpd/conf.d/ssl.conf
SSLCertificateFile /etc/pki/tls/certs/vbird.crt    &lt;==約在 105 行
SSLCertificateKeyFile /etc/pki/tls/certs/vbird.key &lt;==約在 112 行

[root@www ~]# /etc/init.d/httpd restart
```

然後再以瀏覽器去瀏覽 https:// 的網址，就能夠查閱到剛剛建立的憑證資料。不過，因爲我們之前已經有瀏覽過預設的憑證， 所以網頁以及憑證都有被快取過！因此，你可能得需要到瀏覽器的隱私保護的地方，將記錄的憑證刪除，並且將網頁快取刪除， 這樣才能夠看到最終如下的正確憑證資料喔！

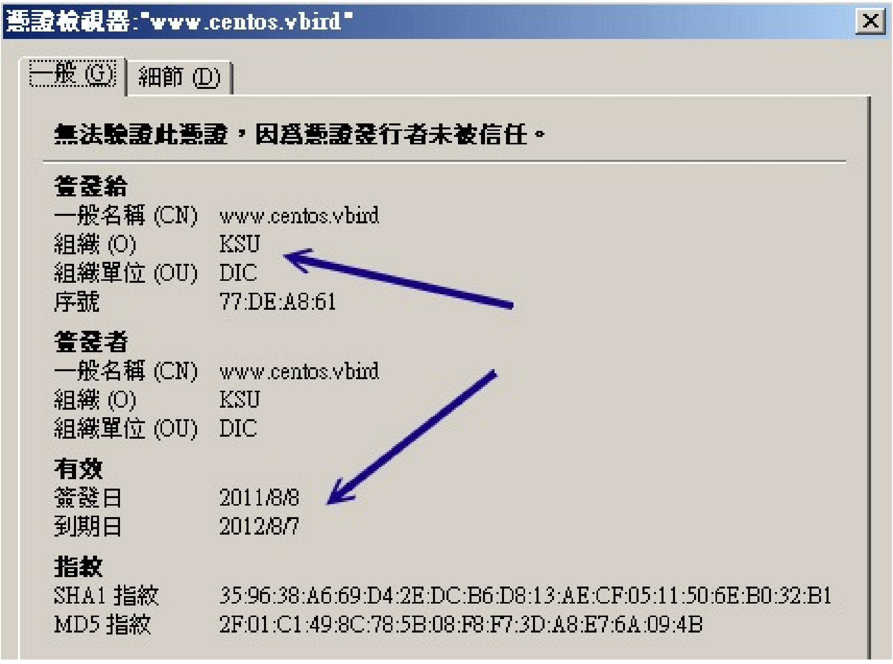


圖 20.5-4、檢查憑證的詳細內容！

20.5.3 將加密首頁與非加密首頁分離

或許你已經發現一個無俚頭的地方，就是我的 http:// 以及 https:// 首頁是一模一樣的嘛！那麼我的讀者幹嘛沒事找事幹， 肯定不會使用 https 的嘛！那怎辦？怎麼強制使用者使用 https:// 來查閱我的重要資料？很簡單啊！ 透過虛擬主機就好了啊！因爲 SSL 模組也是預設提供了這個功能的嘛！修改會不會很麻煩呢？不會啦！ 你只要將 http 及 https 的首頁分離即可！我們這麼假設好了：

- 一般明碼傳輸的網頁首頁不要變更；
- https:// 的首頁放置到 /var/www/https/ 目錄下。

所以我們得先要設定 /var/www/https 目錄才行！然後，只要修改 ssl.conf 檔案內容即可！整個過程可以這樣處理：

```
# 1\. 處理目錄與預設的首頁 index.html 檔案：
[root@www ~]# mkdir /var/www/https
[root@www ~]# echo "This is https' home" &gt; /var/www/https/index.html

# 2\. 開始處理 ssl.conf 的內容囉！
[root@www ~]# vim /etc/httpd/conf.d/ssl.conf
Listen 443                      &lt;==預設的監聽埠口！不建議修改！
&lt;VirtualHost _default_:443&gt;     &lt;==就是虛擬主機的設定囉！
DocumentRoot "/var/www/https"   &lt;==約84行，拿掉註解改掉目錄名稱
ServerName *:443                &lt;==拿掉註解，並將主機名稱設定爲 *
SSLEngine on                    &lt;==有支援 SSL 的意思！
SSLCipherSuite ALL:!ADH:!EXPORT:!SSLv2:RC4+RSA:+HIGH:+MEDIUM:+LOW
SSLCertificateFile /etc/pki/tls/certs/vbird.crt
SSLCertificateKeyFile /etc/pki/tls/certs/vbird.key
&lt;/VirtualHost&gt;

[root@www ~]# /etc/init.d/httpd restart
```

大部分都使用預設值，就是 DocumentRoot 以及 ServerName 需要留意就是了。如此一來，我們就將 https, http 兩個完整的分開，你的重要資料需要加密的，終於有個可靠的地方擺放囉！^_^

20.5.4 防砍站軟體

幾個比較知名的網站管理員大概都有這樣的困擾，那就是網站常被砍站軟體所強力下載，結果造成主機的 CPU loading 過重，最後竟然會導致死掉～唉！眞是的～人怕出名豬怕肥吶！先來解釋一下什麼是砍站吧！

所謂的『砍站』，就是以類似多點連線下載的持續性訊息傳遞軟體進行網站資料的下載，而且， 一啓用該軟體，該軟體就將『整個網站』的內容都給他 download 下來，很厲害吧！沒錯！是很厲害，但是卻也害死人了～怎麼說呢？

因爲這種軟體常常會爲了加快 download 的速度，所以採用多點連線的方式，也就是會持續不斷的向 Server 發出要求封包，而由於這些封包並不見得能夠成功的讓 Server 把資料傳導給 Client 端，常常會無法投遞就是啦！這樣的結果就是...造成 Server 要一直不斷的回應，又無法正確的回應出去， 此外，要求太過頻繁，結果主機應接不暇，最後...就當機了...眞的是林老師ㄉㄟ～

鳥哥的鳥站主機古早以前，就是因爲這樣的原因，導致服務常常斷斷續續的，並且，由於 CPU loading 太高，結果讓正常連線進來看資料的網友沒有足夠的資源，因此網頁開啓的速度就變的很慢～唉～ 這些砍站的人，也太不道德啦！

由於這種砍站軟體眞的很麻煩，一不注意馬上就又會被砍站而當機，三天兩頭就要重新開機一次，完全讓 Linux 的穩定性無法發揮！眞是氣死了～後來，鳥哥就自行寫了一個 scripts 來擋這樣的 IP ！我的作法是這樣的：

1. 由於砍站軟體會多點連續下載，因此，同一個 IP 在同一個時間內，會有相當多的連線發生；

2. 由於他是重複不斷的要求連線，因此剛剛建立的連線在達成下載的目的後，會立刻死掉， 而又多生出其他的連線出來，因此，這個時候他的連線情況就變的相當的不正常了！
3. 由於某些較舊的砍站軟體並不會『欺騙』主機，所以，會在主機的登錄檔裡面記錄住 Teleport 的標記！
4. 既然如此的話，那麼我就讓我的主機每分鐘去檢查兩個東西(1)先檢查 log file ，如果有發現到相關的 Teleport 字詞，就將該 IP 抵擋掉；(2)使用 netstat 來檢查同一個 IP 的同時連線，如果該連線超過一個值(例如同時有 12 個連線)的話，那麼就將該 IP 抵擋掉！
5. 此外，由於上面的方案可能會將 Proxy 的 Client 端也同時抵擋掉，眞是可憐啊！ 這個時候，這支程式就會主動的將(1)的情況的主機抵擋 3 天，至於(2)的情況則抵擋2小時！ 過了該抵擋的時限後，該 IP 即可又連上我們的主機了！

大致上就是這樣吧！這樣的一程式需要與 iptables 相互配合，所以，請先查閱一下第九章的防火牆內容，然後再來下載這支程式吧！ 這支程式你可以在底下的網址下載喔！

- http://linux.vbird.org/download/index.php?action=detail&fileid=47

詳細的安裝步驟鳥哥已經以中文寫在該檔案裡面了，所以請先查看一下該檔案的前面說明部分吧！此外， Study Area 的 netman 大哥也已經開發了一套很棒的防砍站的程式了！ 在防堵砍站的原理上面是完全相同的，不過寫法可能不是很雷同就是了！如果有需要的話，也可以前往 Study-Area 搜尋一下囉！

- http://phorum.study-area.org/viewtopic.php?t=13643

20.6 重點回顧

- WWW 的傳輸協定使用 HTTP (Hyper Text Transport Protocol) ，最早是由歐洲核子物理實驗室的伯納斯-李所發展的；
- WWW 在 server/client 端主要傳遞的訊息資料以 HTML (Hyper Text Markup Language) 語法爲主；
- http://www.w3c.org 爲制訂與發佈 WWW 標準語法的組織，你撰寫網頁最好依據該站之標準爲宜；
- Apache 是達成 WWW 伺服器的一項軟體，至於用户端的瀏覽則使用瀏覽器，目前可使用 firefox
- 瀏覽器可達成的主機連結不止 http，可在網址列輸入對應的『協定://主機[:port]/資源』即可取得不同的資料；
- 若要 WWW 伺服器可以達成與使用者資訊互動，尚須要網頁程式語言 (如PHP, perl 等) 以及資料庫軟體 (如 MySQL, portgresql 等)；
- 因爲 http 使用的是明碼傳送，目前 WWW 可利用 SSL 等機制來進行資料加密的傳輸；
- Apache 的設定檔其實只有 httpd.conf 而已，其他的設定檔都是被 Include 進來的；
- Apache 的首頁目錄以 DocumentRoot 決定，首頁檔案則以 DirectoryIndex 決定；
- Apache 可以透過虛擬主機的設定以指定不同主機名稱到不同的 DocumentRoot 下；
- Apache 是多執行緒的軟體，可以啓動多個程序來負責 WWW。主要的模組有 prefork 及 worker， 至於最大可連線的數量則以 MaxClients 來決定。
- 若要正確的讓瀏覽器顯示網頁的編碼格式，最好在網頁上宣告語系，並將 Apache 的設定檔 httpd.conf 內的 AddDefaultCharset 設定值取消；
- 在 Apache 可瀏覽的目錄權限設定上 (Options 參數)，最好將 Indexes拿掉；
- 透過 AllowOverride 與 .htaccess 可讓使用者在自己管理的目錄下制訂自己的風格；
- Apache 本身提供一個 apachectl 的 script 讓使用者得以快速管理其 apache 的服務；
- Apache 分析的資料如果比較重要時，務必以 SSL 或者是保護目錄來保護。

20.7 本章習題

- 請問 LAMP 這個伺服器代表什麼意思？這個名詞代表了 Linux + Apache + MySQL + PHP 這個 WWW 伺服器的組成！
- Apache 的設定檔檔名一般爲何？Apache 的設定檔檔名爲 httpd.conf ，不過，由於 httpd.conf 內容參數可以使用『 include "額外設定檔名"』，所以也可能具有其他的額外設定檔喔！
- 在 Apache 的設定檔當中，哪一個參數是用來設定『主網頁』的？設定主網頁的參數爲：DocumentRoot 喔！後面接的是主網頁放置的『目錄』！
- 哪一個指令用來重新啓動與關閉 Apache ？(請以 Apache 本身提供的功能來說明)其實不論是 RPM 還是 Tarball 都是使用 apachectl 這個檔案來啓動 apache 的，不過 RPM 已經將該檔案整合到 /etc/init.d/httpd 裡面去而已
- 當我使用 ps -aux 的時候，發現好多的 httpd... 的程序，這是正常的嗎？ 最多可以有幾個程序是在那個檔案的那個參數所設定的？由於 Apache 預設爲多執行緒，所以啓動多個 processes 是正常的。至於啓動幾個 process 則由很多設定所處理， 包括 MinSpareServers, MaxSpareServers, MaxClients 等等。
- 又，呈上題，這些程序 (process) 的 owner 與 group 是誰？該察看那個設定檔的那個參數？同樣察看 httpd.conf 裡面的 User 與 Group 這兩個設定值！
- 如果今天我以 http://your.ip 結果卻發現瀏覽器出現類似 FTP 的畫面(會列出該目錄下的所有檔案)，這是什麼原因造成的？該如何避免？這是由於在 httpd.conf 裡面，針對該目錄的設定參數『 Options 』當中，設定了 Indexes 這個設定值，導致當找不到主頁時(通常是 index.html)，就會將該目錄下的所有檔案秀出來！解決的方法就是拿掉 Options 裡面的 Indexes 設定值即可！
- 在 Apache 裡面 .htaccess 這個檔案的功能爲何？可以用來取代 httpd.conf 裡面的設定參數！創造屬於使用者自己的 Apache 風格！
- 若你之前瀏覽過網頁，但顯示的資料並非正確的中文。後來按照上文的說明修改了中文的設定，卻還是無法看到中文。 請問可能的原因爲何？由於你曾經瀏覽過該網站的網頁，所以該網頁會被你的瀏覽器所暫存 (cache)，因此你應該可以這樣做：
    - 在同一頁面下按下『 reload 』來重新載入；
    - 清除掉所有的瀏覽器快取；
    - 將原本的網頁在伺服器端改名，並讓瀏覽器瀏覽新的網頁名稱。
- PHP 的程式碼一定要使用 <?php 程式碼 ?> 嗎？有沒有替代方案？預設的情況下，你應該要輸入 <?php ?> 才能寫入 PHP 的程式。不過早期的程式或許都以 <? ?> 來撰寫的。如果想要讓該種方式生效的話，你可以進入 /etc/php.ini 檔案中，修改『short_open_tag = On』這個設定項目即可。

20.8 參考資料與延伸閱讀

- 葛林.穆迪著，杜默譯，『Linux傳奇』，時報出版；
- WWW 發展者蒂姆•伯納斯－李的生平簡介： http://zh.wikipedia.org/wiki/蒂姆•伯納斯－李
- W3C 標準制訂與公布網站：http://www.w3c.org
- Apache 官方網站： http://www.apache.org/
- Mozilla 官方網站： http://www.mozilla.org/
- PHP 官方網站： http://www.php.net/
- MySQL 官方網站： http://www.mysql.org/
- MySQL 中文使用手冊： http://linux.tnc.edu.tw/techdoc/mysql/mysql_doc/manual_toc.html
- Apache 1.3 版的 Tarball 安裝方式：http://linux.vbird.org/linux_server/0360apache/0360apache-1.php
- Apache 2.0 的說明文件： http://httpd.apache.org/docs/2.0/mod/core.html
- 林彥明的 Apache SSL 實戰演練：http://www.vbird.org/somepaper/20060310-https.pdf

---

2003/01/14：第一次完成 2003/01/18：新增問題討論：關於中文的說明 2003/01/21：新增問題討論：關於 PHP 無法使用的問題說明 2003/04/28：加入砍站軟體的程式說明 2003/04/29：加入 PHP 原始碼程式優化模組 MM Cache 說明。 2003/05/07：加入 ab 這個效能測試的說明！ 2003/05/30：使用 Tarball 安裝時常常發生一些困擾，加入 User/Group 的設定說明！ 2003/09/10：將原本在 2002/12 安裝 Tarball 的軟體更新爲目前 2003/09 最新的版本來安裝喔！ 2003/10/02：加入一些問題的克服之道喔！ 2004/03/25：修訂2004/03/25：修訂 MySQL 安裝的流程！第四步驟加入權限的修訂！ 2004/09/03：修改了 MMCache 的主網頁。 2006/10/21：將舊的文章移動到 此處 2006/11/09：花了很多時間修改，不再提供 tarball 的安裝需求了！ 2006/11/10：預先釋出版本，包括修改 MM Cache 成爲 eaccelerator、增加 SSL 修改 awstats 之安裝等。 2010/02/08：網友告知，SSL 建置的 genrsa 應該是 private key 而非 public key 喔！這部份鳥哥誤解了。 2011/05/10：將舊的基於 CentOS 4.x 的版本移動到 此處 2011/05/27：終於改完了！這次的改版幅度不會很大，主要是適應在 CentOS 5.x 的版本上面啦！ 2011/08/05：將基於 CentOS 5.x 的版本移動到 此處

第二十一章、文件服务器之三： FTP 服务器

最近更新日期：2011/08/08

FTP (File Transfer Protocol) 可说是最古老的协议之一了，主要是用来进行档案的传输，尤其是大型档案的传输使用 FTP 更是方便！不过，值得注意的是，使用 FTP 来传输时，其实是具有一定程度的『危险性』， 因为数据在因特网上面是完全没有受到保护的『明码』传输方式！但是单纯的 FTP 服务还是有其必要性的，例如很多学校就有 FTP 服务器的架设需求啊！

21.1 FTP 的数据链路原理

FTP (File transfer protocol) 是相当古老的传输协议之一，他最主要的功能是在服务器与客户端之间进行档案的传输。 这个古老的协议使用的是明码传输方式，且过去有相当多的安全危机历史。为了更安全的使用 FTP 协议，我们主要介绍较为安全但功能较少的 vsftpd 这个软件呐。

2.1.1 FTP 功能简介

FTP 服务器的功能除了单纯的进行档案的传输与管理之外，依据服务器软件的设定架构，它还可以提供几个主要的功能。 底下我们约略的来谈一谈：

- 不同等级的用户身份：user, guest, anonymous

FTP 服务器在预设的情况下，依据使用者登入的情况而分为三种不同的身份，分别是： (1)实体账号,real user；(2)访客, guest；(3)匿名登录者, anonymous 这三种。这三种身份的用户在系统上面的权限差异很大喔！例如实体用户取得系统的权限比较完整， 所以可以进行比较多的动作；至于匿名登录者，大概我们就仅提供他下载资源的能力而已，并不许匿名者使用太多主机的资源啊！ 当然，这三种人物能够使用的『在线指令』自然也就不相同啰！ ^_^

- 命令记录与登录文件记录：

FTP 可以利用系统的 syslogd 来进行数据的纪录， 而记录的数据报括了用户曾经下达过的命令与用户传输数据(传输时间、档案大小等等)的纪录呢！ 所以你可以很轻松的在 /var/log/ 里面找到各项登录信息喔！

- 限制用户活动的目录： (change root, 简称 chroot)

为了避免用户在你的 Linux 系统当中随意逛大街 (意指离开用户自己的家目录而进入到 Linux 系统的其他目录去)， 所以将使用者的工作范围『局限』在用户的家目录底下，嗯！实在是个不错的好主意！FTP 可以限制用户仅能在自己的家目录当中活动喔！如此一来，由于使用者无法离开自己的家目录，而且登入 FTP 后，显示的『根目录』就是自己家目录的内容，这种环境称之为 change root ，简称 chroot ，改变根目录的意思啦！

这有什么好处呢？当一个恶意的使用者以 FTP 登入你的系统当中，如果没有 chroot 的环境下，他可以到 /etc, /usr/local, /home 等其他重要目录底下去察看档案数据，尤其是很重要的 /etc/ 底下的配置文件，如 /etc/passwd 等等。如果你没有做好一些档案权限的管理与保护，那他就有办法取得系统的某些重要信息， 用来『入侵』你的系统呢！所以在 chroot 的环境下，当然就比较安全一些咯！

21.1.2 FTP 的运作流程与使用到的端口

FTP 的传输使用的是 TCP 封包协议，在第二章网络基础中我们谈过， TCP 在建立联机前会先进行三向交握。不过 FTP 服务器是比较麻烦一些，因为 FTP 服务器使用了两个联机，分别是命令信道与数据流通道 (ftp-data) 。这两个联机都需要经过三向交握， 因为是 TCP 封包嘛！那么这两个联机通道的关系是如何呢？底下我们先以 FTP 预设的主动式 (active) 联机来作个简略的说明啰：

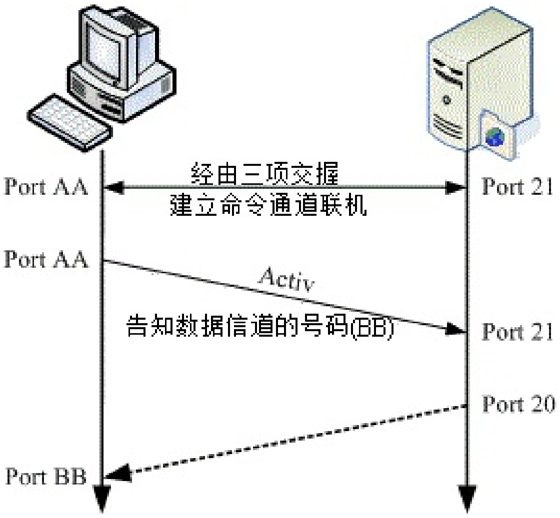


图 21.1-1、FTP 服务器的主动式联机示意图

简单的联机流程就如上图所示，至于联机的步骤是这样的：

1. 建立命令通道的联机 如上图所示，客户端会随机取一个大于 1024 以上的埠口 (port AA) 来与 FTP 服务器端的 port 21 达成联机， 这个过程当然需要三向交握了！达成联机后客户端便可以透过这个联机来对 FTP 服务器下达指令， 包括查询档名、下载、上传等等指令都是利用这个通道来下达的；

2. 通知 FTP 服务器端使用 active 且告知连接的埠号 FTP 服务器的 21 埠号主要用在命令的下达，但是当牵涉到数据流时，就不是使用这个联机了。 客户端在需要数据的情况下，会告知服务器端要用什么方式来联机，如果是主动式 (active) 联机时， 客户端会先随机启用一个埠口 (图 21.1-1 当中的 port BB) ，且透过命令通道告知 FTP 服务器这两个信息，并等待 FTP 服务器的联机；

3. FTP 服务器『主动』向客户端联机 FTP 服务器由命令通道了解客户端的需求后，会主动的由 20 这个埠号向客户端的 port BB 联机， 这个联机当然也会经过三向交握啦！此时 FTP 的客户端与服务器端共会建立两条联机，分别用在命令的下达与数据的传递。 而预设 FTP 服务器端使用的主动联机埠号就是 port 20 啰！

如此一来则成功的建立起『命令』与『数据传输』两个信道！不过，要注意的是， 『数据传输信道』是在有数据传输的行为时才会建立的通道喔！并不是一开始连接到 FTP 服务器就立刻建立的通道呢！留意一下啰！

- 主动式联机使用到的埠号

利用上述的说明来整理一下 FTP 服务器端会使用到的埠号主要有：

- 命令通道的 ftp (默认为 port 21) 与
- 数据传输的 ftp-data (默认为port 20)。

再强调一次，这两个埠口的工作是不一样的，而且，重要的是两者的联机发起端是不一样的！首先 port 21 主要接受来自客户端的主动联机，至于 port 20 则为 FTP 服务器主动联机至客户端呢！这样的情况在服务器与客户端两者同时为公共 IP (Public IP) 的因特网上面通常没有太大的问题，不过，万一你的客户端是在防火墙后端，或者是 NAT 服务器后端呢？会有什么问题发生呢？底下我们来谈一谈这个严重的问题！

- 在主动联机的 FTP 服务器与客户端之间具有防火墙的联机问题

回想一下我们的第九章防火墙！ 一般来说，很多的局域网络都会使用防火墙 (iptables) 的 NAT 功能，那么在 NAT 后端的 FTP 用户如何连接到 FTP 服务器呢？ 我们可以简单的以下图来说明：

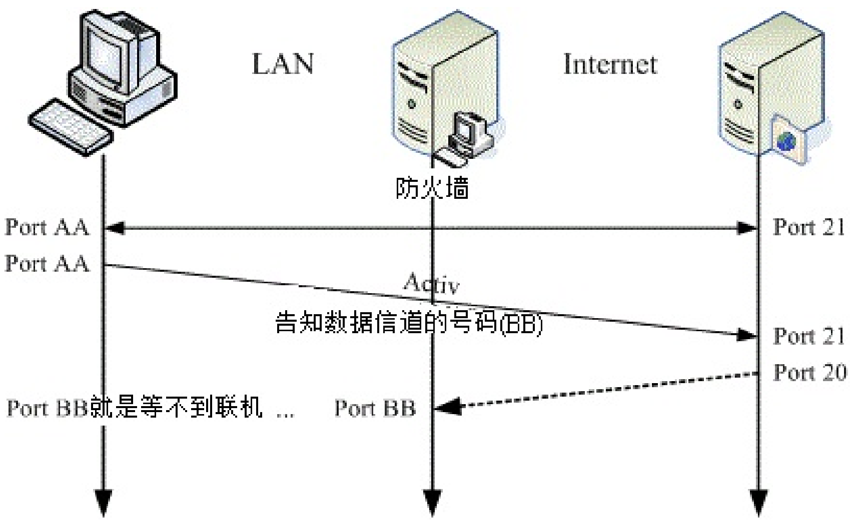


图 21.1-2、 FTP 客户端与服务器端联机中间具有防火墙的联机状态

1. 用户与服务器间命令信道的建立： 因为 NAT 会主动的记录由内部送往外部的联机信息，而由于命令信道的建立是由客户端向服务器端联机的， 因此这一条联机可以顺利的建立起来的；
2. 用户与服务器间数据信道建立时的通知： 同样的，客户端主机会先启用 port BB ，并透过命令通道告知 FTP 服务器，且等待服务器端的主动联机；
3. 服务器主动连到 NAT 等待转递至客户端的联机问题： 但是由于透过 NAT 的转换后，FTP 服务器只能得知 NAT 的 IP 而不是客户端的 IP ， 因此 FTP 服务器会以 port 20 主动的向 NAT 的 port BB 发送主动联机的要求。 但你的 NAT 并没有启动 port BB 来监听 FTP 服务器的联机啊！

了解问题的所在了吗？在 FTP 的主动式联机当中，NAT 将会被视为客户端，但 NAT 其实并非客户端啊， 这就造成问题了。如果你曾经在 IP 分享器后面连接某些 FTP 服务器时，可能偶尔会发现明明就连接上 FTP 服务器了 (命令通道已建立)，但是就是无法取得文件名的列

表，而是在超过一段时间后显示『 Can't build data connection: Connection refused，无法进行数据传输』之类的讯息， 那肯定就是这个原因所造成的困扰了。

那有没有办法可以克服这个问题呢？难道真的在 Linux NAT 后面就一定无法使用 FTP 吗？当然不是！ 目前有两个简易的方法可以克服这个问题：

- 使用 iptables 所提供的 FTP 侦测模块：

  其实 iptables 早就提供了许多好用的模块了，这个 FTP 当然不会被错过！ 你可以使用 modprobe 这个指令来加载 ip*conntrack_ftp* 及 *ip_nat_ftp* 等模块，这几个模块会主动的分析『目标是 *port 21* 的联机』信息， 所以可以得到 *port BB* 的资料，此时若接受到 *FTP* 服务器的主动联机，就能够将该封包导向正确的后端主机了！ ^^

  不过，如果你链接的目标 FTP 服务器他的命令通道默认端口号并非标准的 21 埠号时 (例如某些地下 FTP 服务器)， 那么这两个模块就无法顺利解析出来了，这样说，理解吗？

- 客户端选择被动式 (Passive) 联机模式：

  除了主动式联机之外，FTP 还提供一种称为被动式联机的模式，什么是被动式呢？ 既然主动式是由服务器向客户端联机，反过来讲，被动式就是由客户端向服务器端发起联机的啰！ 既然是由客户端发起联机的，那自然就不需要考虑来自 port 20 的联机啦！关于被动式联机模式将在下一小节介绍喔！

21.1.3 客户端选择被动式联机模式

那么什么是被动式联机呢？我们可以使用底下的图示来作个简略的介绍喔：

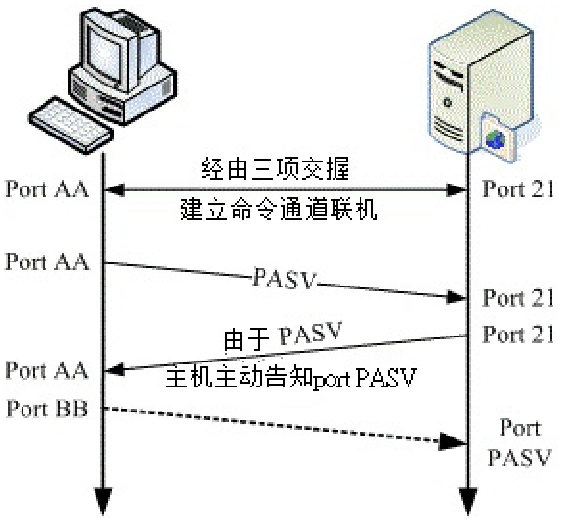


图 21.1-3、FTP 的被动式数据流联机流程

1. 用户与服务器建立命令信道： 同样的需要建立命令通道，透过三向交握就可以建立起这个通道了。

2. 客户端发出 PASV 的联机要求：当有使用数据信道的指令时，客户端可透过命令通道发出 PASV 的被动式联机要求 (Passive 的缩写)，并等待服务器的回应；
3. FTP 服务器启动数据端口，并通知客户端联机：如果你的 FTP 服务器是能够处理被动式联机的，此时 FTP 服务器会先启动一个埠口在监听。这个端口号码可能是随机的，也可以自定义某一范围的埠口，端看你的 FTP 服务器软件而定。然后你的 FTP 服务器会透过命令通道告知客户端该已经启动的埠口 (图中的 port PASV)，并等待客户端的联机。
4. 客户端随机取用大于 1024 的埠口进行连接：然后你的客户端会随机取用一个大于 1024 的端口号来对主机的 port PASV 联机。如果一切都顺利的话，那么你的 FTP 数据就可以透过 port BB 及 port PASV 来传送了。

发现上面的不同点了吗？被动式 FTP 数据信道的联机方向是由客户端向服务器端联机的喔！如此一来，在 NAT 内部的客户端主机就可以顺利的连接上 FTP Server 了！但是，万一 FTP 主机也是在 NAT 后端那怎么办...呵呵！那可就糗了吧～ @_@这里就牵涉到更深入的 DMZ 技巧了，我们这里暂不介绍这些深入的技巧，先理解一下这些特殊的联机方向，这将有助于你未来服务器架设时候的考虑因素喔！

此外，不晓得你有无发现，透过 PASV 模式，服务器在没有特别设定的情况下，会随机选取大于 1024 的埠口来提供客户端连接之用。那么万一服务器启用的埠口被搞鬼怎么办？而且，如此一来也很难追踪来自入侵者攻击的登录信息啊！所以，这个时候我们可以透过 passive ports 的功能来『限定』服务器启用的 port number 喔！

21.1.4 FTP 的安全性问题与替代方案

其实，在 FTP 上面传送的数据很可能被窃取，因为 FTP 是明码传输的嘛！而且某些 FTP 服务器软件的资安历史问题也是很严重的。因此，一般来说，除非是学校或者是一些社团单位要开放没有机密或授权问题的资料之外，FTP 是少用为妙的。

拜 SSH 所赐，目前我们已经有较为安全的 FTP 了，那就是 ssh 提供的 sftp 这个 server 啊！这个 sftp-server 最大的优点就是：『在上面传输的数据是经过加密的』！所以在因特网上面流窜的时候，嘿嘿！毕竟是比较安全一些啦！所以建议你，除非必要，否则的话使用 SSH 提供的 sftp-server 功能即可～

然而这个功能对于一些习惯了图形接口，或者是有中文档名的使用者来说，实在是不怎么方便，虽说目前有个图形接口的 filezilla 客户端软件，不过很多时候还是会发生一些莫名的问题说！所以，有的时候 FTP 网站还是有其存在的需要的。如果真的要架设 FTP 网站，那么还是得需要注意几个事项喔：

1. 随时更新到最新版本的 FTP 软件，并随时注意漏洞讯息；
2. 善用 iptables 来规定可以使用 FTP 的网域；
3. 善用 TCP_Wrappers 来规范可以登入的网域；

4. 善用 FTP 软件的设定来限制使用你 FTP 服务器的使用者的不同权限啊；
5. 使用 Super daemon 来进阶管理你的 FTP 服务器；
6. 随时注意用户的家目录、以及匿名用户登入的目录的『档案权限』；
7. 若不对外公开的话，或许也可以修改 FTP 的 port 。
8. 也可以使用 FTPs 这种加密的 FTP 功能！

无论如何，在网络上听过太多人都是由于开放 FTP 这个服务器而导致整个主机被入侵的事件，所以， 这里真的要给他一直不断的强调，要注意安全啊！

21.1.5 开放什么身份的使用者登入

既然 FTP 是以明码传输，并且某些早期的 FTP 服务器软件也有不少的安全漏洞，那又为何需要架设 FTP 服务器啊？ 没办法啊，总是有人有需要这个玩意儿的，譬如说各大专院校不就有提供 FTP 网站的服务吗？ 这样可以让校内的同学共同分享校内的网络资源嘛！不过，由于 FTP 登入者的身份可以分为三种， 你到底要开放哪一种身份登入呢？这个时候你可以这样简单的思考一下啰：

- 开放实体用户的情况 (Real user)：

很多的 FTP 服务器默认就已经允许实体用户的登入了。不过，需要了解的是，以实体用户做为 FTP 登入者身份时， 系统默认并没有针对实体用户来进行『限制』的，所以他可以针对整个文件系统进行任何他所具有权限的工作。 因此，如果你的 FTP 使用者没能好好的保护自己的密码而导致被入侵，那么你的整个 Linux 系统数据将很有可能被窃取啊！ 开放实体用户时的建议如下：

- 使用替代的 FTP 方案较佳： 由于实体用户本来就可以透过网络连接到主机来进行工作 (例如 SSH)，因此实在没有需要特别的开放 FTP 的服务啊！因为例如 sftp 本来就能达到传输档案的功能啰！

- 限制用户能力，如 chroot 与 /sbin/nologin 等： 如果确定要让实体用户利用 FTP 服务器的话，那么你可能需要让某些系统账号无法登入 FTP 才行，例如 bin, apache 等等。 最简单常用的作法是透过 PAM 模块来处理，譬如 vsftpd 这个软件默认可以透过 /etc/vsftpd/ftpusers 这个档案来设定不想让他具有登入 FTP 的账号。另外，将使用者身份 chroot 是相当需要的！

- 访客身份 (Guest)

通常会建立 guest 身份的案例当中，多半是由于服务器提供了类似『个人 Web 首页』的功能给一般身份用户， 那么这些使用者总是需要管理自己的网页空间吧？这个时候将使用者的身份压缩成为 guest ，并且将他的可用目录设定好，即可提供使用者一个方便的使用环境了！ 且不需要提供他 real user 的权限喔！ 常见的建议如下：

- 仅提供需要登入的账号即可，不需要提供系统上面所有人均可登入的环境啊！

- 当然，我们在服务器的设定当中，需要针对不同的访客给他们不一样的『家目录』，而这个家目录与用户的权限设定需要相符合喔！例如要提供 dmtsai 这个人管理他的网页空间，而他的网页空间放置在 /home/dmtsai/www 底下，那我就将 dmtsai 在 FTP 提供的目录仅有 /home/dmtsai/www 而已，比较安全啦！而且也方便使用者啊！
- 针对这样的身份者，需要设定较多的限制，包括：上下传档案数目与硬盘容量的限制、联机登入的时间限制、许可使用的指令要减少很多很多，例如 chmod 就不要允许他使用等等！
- 匿名登录使用者 (anonymous)

虽然提供匿名登录给因特网的使用者进入实在不是个好主意，因为每个人都可以去下载你的数据， 万一带宽被吃光光怎么办？但如同前面讲过的，学校单位需要分享全校同学一些软件资源时， FTP 服务器也是一个很不错的解决方案啊！你说是吧。如果要开放匿名用户的话，要注意：

- 无论如何，提供匿名登录都是一件相当危险的事情，因为只要你一不小心， 将重要的资料放置到匿名者可以读取的目录中时，那么就很有可能会泄密！与其战战兢兢，不如就不要设定啊～
- 果真要开放匿名登录时，很多限制都要进行的，这包括：(1)允许的工作指令要减低很多， 几乎就不许匿名者使用指令啦、(2)限制文件传输的数量，尽量不要允许『上传』数据的设定、 (3)限制匿名者同时登入的最大联机数量，可以控制盗连喔！

一般来说，如果你是要放置一些公开的、没有版权纠纷的数据在网络上供人下载的话， 那么一个仅提供匿名登录的 FTP 服务器，并且对整个因特网开放是 OK 的啦！ 不过，如果你预计要提供的的软件或数据是具有版权的，但是该版权允许你在贵单位内传输的情况下， 那么架设一个『仅针对内部开放的匿名 FTP 服务器 (利用防火墙处理) 』也是 OK 的啦！

如果你还想要让使用者反馈的话，那是否要架设一个匿名者可上传的区域呢？鸟哥对这件事情的看法是....『万万不可』啊！如果要让使用者反馈的话，除非该使用者是你信任的，否则不要允许对方上传！ 所以此时一个文件系统权限管理严格的 FTP 服务器，并提供实体用户的登入就有点需求啦！ 总之，要依照你的需求来思考是否有需要喔！

21.2 vsftpd 服务器基础设定

终于要来聊一聊这个简单的 vsftpd 啰！vsftpd 的全名是『Very Secure FTP Daemon 』的意思， 换句话说，vsftpd 最初发展的理念就是在建构一个以安全为重的 FTP 服务器呢！我们先来聊一聊为什么 vsftpd 号称『非常安全』呢？然后再来谈设定吧！

21.2.1 为何使用 vsftpd

为了建构一个安全为主的 FTP 服务器， vsftpd 针对操作系统的『程序的权限 (privilege)』概念来设计， 如果你读过基础篇的十七章程序与资源管理的话， 应该会晓得系统上面所执行的程序都会引发一个程序，我们称他为 PID (Process ID)， 这个 PID 在系统上面能进行的任务与他拥有的权限有关。也就是说， PID 拥有的权限等级越高， 他能够进行的任务就越多。举例来说，使用 root 身份所触发的 PID 通常拥有可以进行任何工作的权限等级。

不过，万一触发这个 PID 的程序 (program) 有漏洞而导致被网络怪客 (cracker) 所攻击而取得此 PID 使用权时， 那么网络怪客将会取得这个 PID 拥有的权限呐！所以，近来发展的软件都会尽量的将服务取得的 PID 权限降低，使得该服务即使不小心被入侵了，入侵者也无法得到有效的系统管理权限，这样会让我们的系统较为安全的啦。 vsftpd 就是基于这种想法而设计的。

除了 PID 方面的权限之外， vsftpd 也支持 chroot 这个函式的功能，chroot 顾名思义就是『change root directory 』的意思，那个 root 指的是『根目录』而非系统管理员。 他可以将某个特定的目录变成根目录，所以与该目录没有关系的其他目录就不会被误用了。

举例来说，如果你以匿名身份登入我们的 ftp 服务的话，通常你会被限定在 /var/ftp 目录下工作， 而你看到的根目录其实就只是 /var/ftp ，至于系统其他如 /etc, /home, /usr... 等其他目录你就看不到了！ 这样一来即使这个 ftp 服务被攻破了，没有关系，入侵者还是仅能在 /var/ftp 里面跑来跑去而已，而无法使用 Linux 的完整功能。自然我们的系统也就会比较安全啦！

vsftpd 是基于上面的说明来设计的一个较为安全的 FTP 服务器软件，他具有底下的特点喔：

- vsftpd 这个服务的启动者身份为一般用户，所以对于 Linux 系统的权限较低，对于 Linux 系统的危害就相对的减低了。此外， vsftpd 亦利用 chroot() 这个函式进行改换根目录的动作，使得系统工具不会被 vsftpd 这支服务所误用；
- 任何需要具有较高执行权限的 vsftpd 指令均以一支特殊的上层程序所控制， 该上层程序享有的较高执行权限功能已经被限制的相当的低，并以不影响 Linux 本身的系统为准；
- 绝大部分 ftp 会使用到的额外指令功能 (dir, ls, cd ...) 都已经被整合到 vsftpd 主程序当中了，因此理论上 vsftpd 不需要使用到额外的系统提供的指令，所以在 chroot 的情况下， vsftpd 不但可以顺利运作，且不需要额外功能对于系统来说也比较安全。

- 所有来自客户端且想要使用这支上层程序所提供的较高执行权限之 vsftpd 指令的需求，均被视为『不可信任的要求』来处理，必需要经过相当程度的身份确认后，方可利用该上层程序的功能。 例如 chown(), Login 的要求等等动作；
- 此外，上面提到的上层程序中，依然使用 chroot() 的功能来限制用户的执行权限。

由于具有这样的特点，所以 vsftpd 会变的比较安全一些咯！底下就开始来谈如何设定吧！

21.2.2 所需要的软件以及软件结构

vsftpd 所需要的软件只有一个，那就是 vsftpd 啊！^_^！如果你的 CentOS 没有安装，请利用 yum install vsftpd 来安装他吧！软件很小，下载连同安装不需要几秒钟就搞定了！而事实上整个软件提供的配置文件也少的令人高兴！简单易用就是 vsftpd 的特色啊！这些设定数据比较重要的有：

- /etc/vsftpd/vsftpd.conf 严格来说，整个 vsftpd 的配置文件就只有这个档案！这个档案的设定是以 bash 的变量设定相同的方式来处理的， 也就是『参数=设定值』来设定的，注意， 等号两边不能有空白喔！至于详细的 vsftpd.conf 可以使用 『 man 5 vsftpd.conf 』来详查。
- /etc/pam.d/vsftpd 这个是 vsftpd 使用 PAM 模块时的相关配置文件。主要用来作为身份认证之用，还有一些用户身份的抵挡功能， 也是透过这个档案来达成的。你可以察看一下该档案：

```
[root@www ~]# cat /etc/pam.d/vsftpd
#%PAM-1.0
session optional pam_keyinit.so    force revoke
&lt;u&gt;auth    required pam_listfile.so item=user sense=deny file=/etc/vsftpd/ftpus
auth     required pam_shells.so
auth     include  password-auth
account include  password-auth
session required pam_loginuid.so
session include  password-auth
```

上面那个 file 后面接的档案是『限制使用者无法使用 vsftpd 』之意， 也就是说，其实你的限制档案不见得要使用系统默认值，也可以在这个档案里面进行修改啦！ ^_^

- /etc/vsftpd/ftpusers 与上一个档案有关系，也就是 PAM 模块 (/etc/pam.d/vsftpd) 所指定的那个无法登入的用户配置文件啊！ 这个档案的设定很简单，你只要将『不想让他登入 FTP 的账号』写入这个档案即可。一行一个账号，看起来像这样：

```
[root@www ~]# cat /etc/vsftpd/ftpusers
# Users that are not allowed to login via ftp
root
bin
daemon
....(底下省略)....
```

瞧见没有？绝大部分的系统账号都在这个档案内喔，也就是说，系统账号默认是没有办法使用 vsftpd 的啦！ 如果你还想要让某些使用者无法登入，写在这里是最快的！

- /etc/vsftpd/user_list 这个档案是否能够生效与 vsftpd.conf 内的两个参数有关，分别是『 userlist_enable, userlist_deny 』。 如果说 /etc/vsftpd/ftpusers 是 PAM 模块的抵挡设定项目，那么这个 /etc/vsftpd/user_list 则是 vsftpd 自定义的抵挡项目。事实上这个档案与 /etc/vsftpd/ftpusers 几乎一模一样， 在预设的情况下，你可以将不希望可登入 vsftpd 的账号写入这里。不过这个档案的功能会依据 vsftpd.conf 配置文件内的 userlist_deny={YES/NO} 而不同，这得要特别留意喔！
- /etc/vsftpd/chroot_list 这个档案预设是不存在的，所以你必须要手动自行建立。这个档案的主要功能是可以将某些账号的使用者 chroot 在他们的家目录下！但这个档案要生效与 vsftpd.conf 内的『 chroot_list_enable, chroot_list_file 』两个参数有关。 如果你想要将某些实体用户限制在他们的家目录下而不许到其他目录去，可以启动这个设定项目喔！
- /usr/sbin/vsftpd 这就是 vsftpd 的主要执行档咯！不要怀疑， vsftpd 只有这一个执行档而已啊！
- /var/ftp/ 这个是 vsftpd 的预设匿名者登入的根目录喔！其实与 ftp 这个账号的家目录有关啦！

大致上就只有这几个档案需要注意而已，而且每个档案的设定又都很简单！真是不错啊！

21.2.3 vsftpd.conf 设定值说明

事实上，/etc/vsftpd/vsftpd.conf 本身就是一个挺详细的配置文件，且使用『 man 5 vsftpd.conf 』则可以得到完整的参数说明。 不过我们这里依旧先将 vsftpd.conf 内的常用参数给他写出来，希望对你有帮助：

- 与服务器环境较相关的设定值
- connect_from_port_20=YES (NO) 记得在前一小节提到的主动式联机使用的 FTP 服务器的 port 吗？这就是 ftp-data 的埠号；
- listen_port=21 vsftpd 使用的命令通道 port，如果你想要使用非正规的埠号，在这个设定项目修改吧！ 不过你必须要知道，这个设定值仅适合以 stand alone 的方式来启动喔！(对于 super daemon 无效)

- dirmessage_enable=YES (NO) 当用户进入某个目录时，会显示该目录需要注意的内容，显示的档案默认是 .message ，你可以使用底下的设定项目来修订！
- message_file=.message 当 dirmessage_enable=YES 时，可以设定这个项目来让 vsftpd 寻找该档案来显示讯息！
- listen=YES (NO) 若设定为 YES 表示 vsftpd 是以 standalone 的方式来启动的！预设是 NO 呦！所以我们的 CentOS 将它改为 YES 哩！这样才能使用 stand alone 的方式来唤醒。
- pasv_enable=YES (NO) 支持数据流的被动式联机模式(passive mode)，一定要设定为 YES 的啦！
- use_localtime=YES (NO) 是否使用本地时间？vsftpd 预设使用 GMT 时间(格林威治)，所以预设的 FTP 内的档案日期会比台湾晚 8 小时，建议修改设定为 YES 吧！
- write_enable=YES (NO) 如果你允许用户上传数据时，就要启动这个设定值；
- connect_timeout=60 单位是秒，在数据连接的主动式联机模式下，我们发出的连接讯号在 60 秒内得不到客户端的响应，则不等待并强制断线咯。
- accept_timeout=60 当用户以被动式 PASV 来进行数据传输时，如果服务器启用 passive port 并等待 client 超过 60 秒而无回应， 那么就给他强制断线！这个设定值与 connect_timeout 类似，不过一个是管理主动联机，一个管理被动联机。
- data_connection_timeout=300 如果服务器与客户端的数据联机已经成功建立 (不论主动还是被动联机)，但是可能由于线路问题导致 300 秒内还是无法顺利的完成数据的传送，那客户端的联机就会被我们的 vsftpd 强制剔除！
- idle_session_timeout=300 如果使用者在 300 秒内都没有命令动作，强制脱机！避免占着茅坑不拉屎～
- max_clients=0 如果 vsftpd 是以 stand alone 方式启动的，那么这个设定项目可以设定同一时间，最多有多少 client 可以同时连上 vsftpd 哩！限制使用 FTP 的用量！
- max_per_ip=0 与上面 max_clients 类似，这里是同一个 IP 同一时间可允许多少联机？
- pasv_min_port=0, pasv_max_port=0 上面两个是与 passive mode 使用的 port number 有关，如果你想要使用 65400 到 65410 这 11 个 port 来进行被动式联机模式的连接，可以这样设定 pasv_max_port=65410 以及 pasv_min_port=65400。 如果是 0 的话，表示随机取用而不限制。
- ftpd_banner=一些文字说明 当使用者联机进入到 vsftpd 时，在 FTP 客户端软件上头会显示的说明文字。不过，这个设定值数据比较少啦！ 建议你可以使用底下的 banner_file 设定值来取代这个项目；

- banner_file=/path/file 这个项目可以指定某个纯文本档作为使用者登入 vsftpd 服务器时所显示的欢迎字眼。同时，也能够放置一些让使用者知道本 FTP 服务器的目录架构！
- 与实体用户较相关的设定值
- guest_enable=YES (NO) 若这个值设定为 YES 时，那么任何实体账号，均会被假设成为 guest 喔 (所以预设是不开放的)！ 至于访客在 vsftpd 当中，预设会取得 ftp 这个使用者的相关权限。但可以透过 guest_username 来修改。
- guest_username=ftp 在 guest_enable=YES 时才会生效，指定访客的身份而已。
- local_enable=YES (NO) 这个设定值必须要为 YES 时，在 /etc/passwd 内的账号才能以实体用户的方式登入我们的 vsftpd 服务器喔！
- local_max_rate=0 实体用户的传输速度限制，单位为 bytes/second， 0 为不限制。
- chroot_local_user=YES (NO) 在预设的情况下，是否要将使用者限制在自己的家目录之内(chroot)？如果是 YES 代表用户默认就会被 chroot，如果是 NO， 则预设是没有 chroot。不过，实际还是需要底下的两个参数互相参考才行。为了安全性，这里应该要设定成 YES 才好。
- chroot_list_enable=YES (NO) 是否启用 chroot 写入列表的功能？与底下的 chroot_list_flie 有关！这个项目得要开启，否则底下的列表档案会无效。
- chroot_list_file=/etc/vsftpd.chroot_list 如果 chroot_list_enable=YES 那么就可以设定这个项目了！这个项目与 chroot_local_user 有关，详细的设定状态请参考 21.2.6 chroot 的说明。
- userlist_enable=YES (NO) 是否藉助 vsftpd 的抵挡机制来处理某些不受欢迎的账号，与底下的参数设定有关；
- userlist_deny=YES (NO) 当 userlist_enable=YES 时才会生效的设定，若此设定值为 YES 时，则当使用者账号被列入到某个档案时， 在该档案内的使用者将无法登入 vsftpd 服务器！该档案文件名与下列设定项目有关。
- userlist_file=/etc/vsftpd/user_list 若上面 userlist_deny=YES 时，则这个档案就有用处了！在这个档案内的账号都无法使用 vsftpd 喔！
- 匿名者登入的设定值
- anonymous_enable=YES (NO) 设定为允许 anonymous 登入我们的 vsftpd 主机！预设是 YES ，底下的所有相关设定都需要将这个设定为 anonymous_enable=YES 之后才会生效！
- anon_world_readable_only=YES (NO) 仅允许 anonymous 具有下载可读档案的权限，预设是 YES。

- anon_other_write_enable=YES (NO) 是否允许 anonymous 具有除了写入之外的权限？包括删除与改写服务器上的档案及档名等权限。预设当然是 NO！如果要设定为 YES，那么开放给 anonymous 写入的目录亦需要调整权限，让 vsftpd 的 PID 拥有者可以写入才行！

- anon_mkdir_write_enable=YES (NO) 是否让 anonymous 具有建立目录的权限？默认值是 NO！如果要设定为 YES， 那么 anony_other_write_enable 必须设定为 YES ！

- anon_upload_enable=YES (NO) 是否让 anonymous 具有上传数据的功能，默认是 NO，如果要设定为 YES ，则 anon_other_write_enable=YES 必须设定。

- deny_email_enable=YES (NO) 将某些特殊的 email address 抵挡住，不让那些 anonymous 登入！如果以 anonymous 登入服务器时，不是会要求输入密码吗？密码不是要你输入你的 email address 吗？如果你很讨厌某些 email address， 就可以使用这个设定来将他取消登入的权限！需与下个设定项目配合：

- banned_email_file=/etc/vsftpd/banned_emails 如果 deny_email_enable=YES 时，可以利用这个设定项目来规定哪个 email address 不可登入我们的 vsftpd 喔！在上面设定的档案内，一行输入一个 email address 即可！

- no_anon_password=YES (NO) 当设定为 YES 时，表示 anonymous 将会略过密码检验步骤，而直接进入 vsftpd 服务器内喔！所以一般预设都是 NO 的！(登入时会检查输入的 emai)

- anon_max_rate=0 这个设定值后面接的数值单位为 bytes/秒 ，限制 anonymous 的传输速度，如果是 0 则不限制(由最大带宽所限制)，如果你想让 anonymous 仅有 30 KB/s 的速度，可以设定『anon_max_rate=30000』

- anon_umask=077 限制 anonymous 上传档案的权限！如果是 077 则 anonymous 传送过来的档案权限会是 -rw------- 喔！

- 关于系统安全方面的一些设定值

- ascii_download_enable=YES (NO) 如果设定为 YES ，那么 client 就优先 (预设) 使用 ASCII 格式下载文件。

- ascii_upload_enable=YES (NO) 与上一个设定类似的，只是这个设定针对上传而言！预设是 NO

- one_process_model=YES (NO) 这个设定项目比较危险一点～当设定为 YES 时，表示每个建立的联机都会拥有一支 process 在负责，可以增加 vsftpd 的效能。不过， 除非你的系统比较安全，而且硬件配备比较高，否则容易耗尽系统资源喔！一般建议设定为 NO 的啦！

- tcp_wrappers=YES (NO) 当然我们都习惯支持 TCP Wrappers 的啦！所以设定为 YES 吧！

- xferlog_enable=YES (NO) 当设定为 YES 时，使用者上传与下载文件都会被纪录起来。记录的档案与下一个设定项目有关：

- xferlog_file=/var/log/xferlog 如果上一个 xferlog_enable=YES 的话，这里就可以设定了！这个是登录档的档名啦！

- xferlog_std_format=YES (NO) 是否设定为 wu ftp 相同的登录档格式？预设为 NO ，因为登录档会比较容易读！ 不过，如果你有使用 wu ftp 登录文件的分析软件，这里才需要设定为 YES

- dual_log_enable=YES, vsftpd_log_file=/var/log/vsftpd.log 除了 /var/log/xferlog 的 wu-ftp 格式登录档之外，还可以具有 vsftpd 的独特登录档格式喔！如果你的 FTP 服务器并不是很忙碌， 或许订出两个登录档的撰写 (/var/log/{vsftpd.log,xferlog) 是不错的。

- nopriv_user=nobody 我们的 vsftpd 预设以 nobody 作为此一服务执行者的权限。因为 nobody 的权限相当的低，因此即使被入侵，入侵者仅能取得 nobody 的权限喔！

- pam_service_name=vsftpd 这个是 pam 模块的名称，我们放置在 /etc/pam.d/vsftpd 即是这个咚咚！

上面这些是常见的 vsftpd 的设定参数，还有很多参数我没有列出来，你可以使用 man 5 vsftpd.conf 查阅喔！不过，基本上上面这些参数已经够我们设定 vsftpd 啰。

21.2.4 vsftpd 启动的模式

vsftpd 可以使用 stand alone 或 super daemon 的方式来启动，我们 CentOS 预设是以 stand alone 来启动的。 那什么时候应该选择 stand alone 或者是 super daemon 呢？如果你的 ftp 服务器是提供给整个因特网来进行大量下载的任务，例如各大专院校的 FTP 服务器，那建议你使用 stand alone 的方式， 服务的速度上会比较好。如果仅是提供给内部人员使用的 FTP 服务器，那使用 super daemon 来管理即可啊。

- 利用 CentOS 提供的 script 来启动 vsftpd (stand alone)

其实 CentOS 不用作任何设定就能够启动 vsftpd 啰！是这样启动的啦：

```
[root@www ~]# /etc/init.d/vsftpd start
[root@www ~]# netstat -tulnp&#124; grep 21
tcp  0  0 0.0.0.0:21  0.0.0.0:*   LISTEN   11689/vsftpd
# 看到啰，是由 vsftpd 所启动的呢！
```

- 自行设定以 super daemon 来启动 (有必要再进行，不用实作)

如果你的 FTP 是很少被使用的，那么利用 super daemon 来管理不失为一个好主意。 不过若你想要使用 super daemon 管理的话，那就得要自行修改一下配置文件了。其实也不难啦，你应该要这样处理的：

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 找到 listen=YES 这一行：大约在 109 行左右啦，并将它改成：
listen=NO
```

接下来修改一下 super daemon 的配置文件，底下这个档案你必须要自行建立的，原本是不存在的喔：

```
[root@www ~]# yum install xinetd   &lt;==假设 xinetd 没有安装时
[root@www ~]# vim /etc/xinetd.d/vsftpd
service ftp
{
        socket_type             = stream
        wait                    = no
        user                    = root
        server                  = /usr/sbin/vsftpd
        log_on_success          += DURATION USERID
        log_on_failure          += USERID
        nice                    = 10
        disable                 = no
}
```

然后尝试启动看看呢：

```
[root@www ~]# /etc/init.d/vsftpd stop
[root@www ~]# /etc/init.d/xinetd restart
[root@www ~]# netstat -tulnp&#124; grep 21
tcp  0  0 0.0.0.0:21  0.0.0.0:*   LISTEN   32274/xinetd
```

有趣吧！两者启动的方式可不一样啊！管理的方式就会差很多的呦！不管你要使用哪种启动的方式，切记不要两者同时启动，否则会发生错误的！你应该使用 chkconfig --list 检查一下这两种启动的方式，然后依据你的需求来决定用哪一种方式启动。鸟哥底下的设定都会以 stand alone 这个 CentOS 默认的启动模式来处理，所以赶紧将刚刚的动作给他改回来喔！

21.2.5 CentOS 的 vsftpd 默认值

在 CentOS 的默认值当中，vsftpd 是同时开放实体用户与匿名用户的，CentOS 的默认值如下：

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 1\. 与匿名者有关的信息：
anonymous_enable=YES        &lt;==支持匿名者的登入使用 FTP 功能

# 2\. 与实体用户有关的设定
local_enable=YES            &lt;==支持本地端的实体用户登入
write_enable=YES            &lt;==允许用户上传数据 (包括档案与目录)
local_umask=022             &lt;==建立新目录 (755) 与档案 (644) 的权限

# 3\. 与服务器环境有关的设定
dirmessage_enable=YES       &lt;==若目录下有 .message 则会显示该档案的内容
xferlog_enable=YES          &lt;==启动登录文件记录，记录于 /var/log/xferlog
connect_from_port_20=YES    &lt;==支持主动式联机功能
xferlog_std_format=YES      &lt;==支持 WuFTP 的登录档格式
listen=YES                  &lt;==使用 stand alone 方式启动 vsftpd
pam_service_name=vsftpd     &lt;==支持 PAM 模块的管理
userlist_enable=YES         &lt;==支持 /etc/vsftpd/user_list 档案内的账号登入管控！
tcp_wrappers=YES            &lt;==支持 TCP Wrappers 的防火墙机制
```

上面各项设定值请自行参考 21.2.3 的详细说明吧。而通过这样的设定值咱们的 vsftpd 可以达到如下的功能：

- 你可以使用 anonymous 这个匿名账号或其他实体账号 (/etc/passwd) 登入；
- anonymous 的家目录在 /var/ftp ，且无上传权限，亦已经被 chroot 了；
- 实体用户的家目录参考 /etc/passwd，并没有被 chroot，可前往任何有权限可进入的目录中；
- 任何于 /etc/vsftpd/ftpusers 内存在的账号均无法使用 vsftpd (PAM)；
- 可利用 /etc/hosts.{allow|deny} 来作为基础防火墙；
- 当客户端有任何上传/下载信息时，该信息会被纪录到 /var/log/xferlog 中；
- 主动式联机的埠口为 port 20；
- <u>使用格林威治时间 (GMT)。</u>

所以当你启动 vsftpd 后，你的实体用户就能够直接利用 vsftpd 这个服务来传输他自己的数据了。 不过比较大的问题是，因为 vsftpd 预设使用 GMT 时间，因为你在客户端使用 ftp 软件连接到 FTP 服务器时，会发现每个档案的时间都慢了八小时了！真是讨厌啊！ 所以建议你加设一个参数值，就是『 use_localtime=YES 』啰！

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 在这个档案当中的最后一行加入这一句即可
use_localtime=YES

[root@www ~]# /etc/init.d/vsftpd restart
[root@www ~]# chkconfig vsftpd on
```

如此一来你的 FTP 服务器不但可以提供匿名账号来下载 /var/ftp 的数据，如果使用实体账号来登入的话， 就能够进入到该用户的家目录底下去了！真是很简单方便的一个设定啊！且使用本地端时间呢！ ^_^

另外，如果你预计要将 FTP 开放给 Internet 使用时，请注意得要开放防火墙喔！关于防火墙的建置情况， 由于牵涉到数据流的主动、被动联机方式，因此，还得要加入防火墙模块。这部份我们在后续的 21.2.8 小节再加以介绍，反正，最终记得要开放 FTP 的联机要求就对了！

21.2.6 针对实体账号的设定

虽然在 CentOS 的默认情况当中实体用户已经可以使用 FTP 的服务了，不过我们可能还需要一些额外的功能来限制实体用户。 举例来说，限制用户无法离开家目录 (chroot)、限制下载速率、限制用户上传档案时的权限 (mask) 等等。 底下我们先列出一些希望达到的功能，然后再继续进行额外功能的处理：

- 希望使用台湾本地时间取代 GMT 时间；
- 用户登入时显示一些欢迎讯息的信息；
- 系统账号不可登入主机 (亦即 UID 小于 500 以下的账号)；
- 一般实体用户可以进行上传、下载、建立目录及修改档案等动作；
- 用户新增的档案、目录之 umask 希望设定为 002；
- 其他主机设定值保留默认值即可。

你可以自行处理 vsftpd.conf 这个档案，以下则是一个范例。注意，如果你的 vsftpd.conf 没有相关设定值， 请自行补上吧！OK！让我们开始一步一步来依序处理先：

1. 先建立主配置文件 vsftpd.conf，这个配置文件已经包含了主要设定值：

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 1\. 与匿名者相关的信息，在这个案例中将匿名登录取消：
anonymous_enable=NO

# 2\. 与实体用户相关的信息：可写入，且 umask 为 002 喔！
local_enable=YES
write_enable=YES
local_umask=002
userlist_enable=YES
userlist_deny=YES
userlist_file=/etc/vsftpd/user_list  &lt;==这个档案必须存在！还好，预设有此档案！

# 3\. 与服务器环境有关的设定
use_localtime=YES
dirmessage_enable=YES
xferlog_enable=YES
connect_from_port_20=YES
xferlog_std_format=YES
listen=YES
pam_service_name=vsftpd
tcp_wrappers=YES
banner_file=/etc/vsftpd/welcome.txt &lt;==&lt;u&gt;这个档案必须存在！需手动建立！&lt;/u&gt;

[root@www ~]# /etc/init.d/xinetd restart  &lt;==取消 super dameon
[root@www ~]# /etc/init.d/vsftpd restart
```

2. 建立欢迎讯息：

   当我们想让登入者可查阅咱们系统管理员所下达的『公告』事项时，可以使用这个设定！那就是 banner_file=/etc/vsftpd/welcome.txt 这个参数的用途了！我们可以编辑这个档案即可。 好了，开始来建立欢迎画面吧！

```
[root@www ~]# vim /etc/vsftpd/welcome.txt
欢迎光临本小站，本站提供 FTP 的相关服务！
主要的服务是针对本机实体用户提供的，
若有任何问题，请与鸟哥联络！
```

3. 建立限制系统账号登入的档案

   再来是针对系统账号来给予抵挡的机制，其实有两个档案啦，一个是 PAM 模块管的，一个是 vsftpd 主动提供的， 在预设的情况下这两个档案分别是：

   - /etc/vsftpd/ftpusers：就是 /etc/pam.d/vsftpd 这个档案的设定所影响的；
   - /etc/vsftpd/user_list：由 vsftpd.conf 的 userlist_file 所设定。 这两个档案的内容是一样的哩～并且这两个档案必须要存在才行。请你参考你的 /etc/passwd 配置文件，然后将 UID 小于 500 的账号名称给他同时写到这两个档案内吧！一行一个账号！

```
[root@www ~]# vim /etc/vsftpd/user_list
root
bin
....(底下省略)....
```

4. 测试结果：

   你可以使用图形接口的 FTP 客户端软件来处理，也可以透过 Linux 本身提供的 ftp 客户端功能哩！ 关于 ftp 指令我们已经在第五章谈过了，你可以自行前往参考。这里直接测试一下吧：

```
# 测试使用已知使用者登入，例如 dmtsai 这个实体用户：
[root@www ~]# ftp localhost
Trying 127.0.0.1...
Connected to localhost (127.0.0.1).
220-欢迎光临本小站，本站提供 FTP 的相关服务！   &lt;==刚刚建立的欢迎讯息
220-主要的服务是针对本机实体用户提供的，
220-若有任何问题，请与鸟哥联络！
220
Name (localhost:root): student
331 Please specify the password.
Password:  &lt;==输入密码啰在这里！
500 OOPS: cannot change directory:/home/student  &lt;==有讲登入失败的原因喔！
Login failed.
ftp&gt; bye
221 Goodbye.
```

   由于默认一般用户无法登入 FTP 的！因为 SELinux 的问题啦！请参考下个小节的方式来处理。 然后以上面的方式测试完毕后，你可以在登入者账号处分别填写 (1)root (2)anonymous 来尝试登入看看！ 如果不能登入的话，那就是设定 OK 的啦！(root 不能登入是因为 PAM 模块以及 user_list 设定值的关系， 而匿名无法登入，是因为我们 vsftpd.conf 里头就是设定不能用匿名登录嘛！)

上面是最简单的实体账号相关设定。那如果你还想要限制用户家目录的 chroot 或其他如速限等数据，就得要看看底下的特殊设定项目啰。

- 实体账号的 SELinux 议题

在预设的情况下，CentOS 的 FTP 是不允许实体账号登入取得家目录数据的，这是因为 SELinux 的问题啦！ 如果你在刚刚的 ftp localhost 步骤中，在 bye 离开 FTP 之前下达过『 dir 』的话，那你会发现没有任何资料跑出来～ 这并不是你错了，而是 SELinux 不太对劲的缘故。那如何解决呢？这样处理即可：

```
[root@www ~]# getsebool -a &#124; grep ftp
allow_ftpd_anon_write --&gt; off
allow_ftpd_full_access --&gt; off
allow_ftpd_use_cifs --&gt; off
allow_ftpd_use_nfs --&gt; off
ftp_home_dir --&gt; off            &lt;==就是这玩意儿！要设定 on 才行！
....(底下省略)....

[root@www ~]# setsebool -P ftp_home_dir=1
```

这样就搞定啰！如果还有其他可能发生错误的原因，包括档案数据使用 mv 而非使用 cp 导致 SELinux 文件类型无法继承原有目录的类型时，那就请自行查阅 /var/log/messages 的内容吧！通常 SELinux 没有这么难处理的啦！^_^

- 对使用者 (包括未来新增用户) 进行 chroot

在鸟哥接触的一般 FTP 使用环境中，大多数都是要开放给厂商联机来使用的，给自己人使用的机会虽然也有， 不过使用者数量通常比较少一些。所以啰，鸟哥现在都是建议默认让实体用户通通被 chroot， 而允许不必 chroot 的账号才需要额外设定。这样的好处是，新建的账号如果忘记进行 chroot，反正原本就是 chroot， 比较不用担心如果该账号是开给厂商时该怎办的问题。

现在假设我系统里面仅有 vbird 与 dmtsai 两个账号不要被 chroot，其他如 student, smb1... 等账号通通预设是 chroot 的啦，包括未来新增账号也全部预设 chroot！那该如何设定？很简单，三个设定值加上一个额外配置文件就搞定了！步骤如下：

```
# 1\. 修改 vsftpd.conf 的参数值：
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 增加是否设定针对某些使用者来 chroot 的相关设定呦！
chroot_local_user=YES
chroot_list_enable=YES
chroot_list_file=/etc/vsftpd/chroot_list

# 2\. 建立不被 chroot 的使用者账号列表，即使没有任何账号，此档案也是要存在！
[root@www ~]# vim /etc/vsftpd/chroot_list
vbird
dmtsai

[root@www ~]# /etc/init.d/vsftpd restart
```

如此一来，除了 dmtsai 与 vbird 之外的其他可用 FTP 的账号者，通通会被 chroot 在他们的家目录下， 这样对系统比较好啦！接下来，请你自己分别使用有与没有被 chroot 的账号来联机测试看看。

- 限制实体用户的总下载流量 (带宽)

你可不希望带宽被使用者上传/下载所耗尽，而影响咱们服务器的其他正常服务吧？所以限制使用者的传输带宽有时也是需要的！ 假设『我要限制所有使用者的总传输带宽最大可达 1 MBytes/秒 』时，你可以这样做即可：

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 增加底下这一个参数即可：
local_max_rate=1000000  &lt;==记住喔，单位是 bytes/second

[root@www ~]# /etc/init.d/vsftpd restart
```

上述的单位是 Bytes/秒，所以你可以依据你自己的网络环境来限制你的带宽！这样就给他限制好啰！有够容易吧！ 那怎么测试啊？很简单，用本机测试最准！你可以用 dd 做出一个 10MB 的档案放在 student 的家目录下，然后用 root 下达 ftp localhost，并输入 student 的帐密，接下来给他 get 这个新的档案，就能够在最终知道下载的速度啦！

- 限制最大同时上线人数与同一 IP 的 FTP 联机数

如果你有限制最大使用带宽的话，那么你可能还需要限制最大在线人数才行！举例来说，你希望最多只有 10 个人同时使用你的 FTP 的话，并且每个 IP 来源最多只能建立一条 FTP 的联机时，那你可以这样做：

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 增加底下的这两个参数：
max_clients=10
max_per_ip=1

[root@www ~]# /etc/init.d/vsftpd restart
```

这样就搞定了！让你的 FTP 不会人满为患呐！

- 建立严格的可使用 FTP 的账号列表

在预设的环境当中，我们是将『不许使用 FTP 的账号写入 /etc/vsftpd/user_list 档案』，所以没有写入 /etc/vsftpd/user_list 当中的使用者就能够使用 FTP 了！如此一来，未来新增的使用者预设都能够使用 FTP 的服务。 如果换个角度来思考，若我想只让某些人可以使用 FTP 而已，亦即是新增的使用者预设不可使用 FTP 这个服务的话那么应该如何作呢？你需要修改配置文件成为这样：

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 这几个参数必须要修改成这样：
userlist_enable=YES
userlist_deny=NO
userlist_file=/etc/vsftpd/user_list

[root@www ~]# /etc/init.d/vsftpd restart
```

则此时『写入 /etc/vsftpd/user_list 变成可以使用 FTP 的账号』了！ 所以未来新增的使用者如果要能够使用 FTP 的话，就必须要写入 /etc/vsftpd/user_list 才行！ 使用这个机制请特别小心，否则容易搞混掉～

透过这几个简单的设定值，相信 vsftpd 已经可以符合大部分合法 FTP 网站的需求啰！ 更多详细的用法则请参考 man 5 vsftpd.conf 吧！

例题：假设你因为某些特殊需求，所以必须要开放 root 使用 FTP 传输档案，那么你应该要如何处理？答：由于系统账号无法使用 FTP 是因为 PAM 模块与 vsftpd 的内建功能所致，亦即是 /etc/vsftpd/ftpusers 及 /etc/vsftpd/user_list 这两个档案的影响。所以你只要进入这两个档案，并且将 root 那一行批注掉，那 root 就可以使用 vsftpd这个 FTP 服务了。 不过，不建议如此作喔！

21.2.7 仅有匿名登录的相关设定

虽然你可以同时开启实体用户与匿名用户，不过建议你，服务器还是依据需求，针对单一种身份来设定吧！ 底下我们将针对匿名用户来设定，且不开放实体用户 。一般来说，这种设定是给类似大专院校的 FTP 服务器来使用的哩！

- 使用台湾本地的时间，而非 GMT 时间；
- 提供欢迎讯息，说明可提供下载的信息；
- 仅开放 anonymous 的登入，且不需要输入密码；
- 文件传输的速限为 1 Mbytes/second；
- 数据连接的过程 (不是命令通道！) 只要超过 60 秒没有响应，就强制 Client 断线！
- 只要 anonymous 超过十分钟没有动作，就予以断线；
- 最大同时上线人数限制为 50 人，且同一 IP 来源最大联机数量为 5 人；
- 预设的 FTP 匿名者的根目录所在： ftp 账号的家目录

OK！那如何设定呢？首先我们必须要知道的是匿名用户的目录在哪里？ 事实上匿名者默认登入的根目录是以 ftp 这个用户的家目录为主，所以你可以使用『 finger ftp 』来查阅。 咱们的 CentOS 默认的匿名者根目录在 /var/ftp/ 中。且匿名登录者在使用 FTP 服务时，他预设可以使用『 ftp 』 这个使用者身份的权限喔，只是被 chroot 到 /var/ftp/ 目录中就是了。

因为匿名者只会在 /var/ftp/ 当中浏览，所以你必须将要提供给用户下载的数据通通给放置到 /var/ftp/ 去。 假设你已经放置了 linux 的相关目录以及 gnu 的相关软件到该目录中了，那我们可以这样做个假设：

```
[root@www ~]# mkdir /var/ftp/linux
[root@www ~]# mkdir /var/ftp/gnu
```

然后将 vsftpd.conf 的数据清空，重新这样处理他吧：

1. 建立 vsftpd.conf 的设定数据

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 将这个档案的全部内容改成这样：
# 1\. 与匿名者相关的信息：
anonymous_enable=YES
no_anon_password=YES        &lt;==匿名登录时，系统不会检验密码 (通常是email)
anon_max_rate=1000000       &lt;==最大带宽使用为 1MB/s 左右
data_connection_timeout=60  &lt;==数据流联机的 timeout 为 60 秒
idle_session_timeout=600    &lt;==若匿名者发呆超过 10 分钟就断线
max_clients=50              &lt;==最大联机与每个 IP 的可用联机
max_per_ip=5

# 2\. 与实体用户相关的信息，本案例中为关闭他的情况！
local_enable=NO

# 3\. 与服务器环境有关的设定
use_localtime=YES
dirmessage_enable=YES
xferlog_enable=YES
connect_from_port_20=YES
xferlog_std_format=YES
listen=YES
pam_service_name=vsftpd
tcp_wrappers=YES
banner_file=/etc/vsftpd/anon_welcome.txt &lt;==档名有改喔！

[root@www ~]# /etc/init.d/vsftpd restart
```

2. 建立欢迎画面与下载提示讯息

各位亲爱的观众朋友！要注意～在这个案例当中，我们将欢迎讯息设定在 /etc/vsftpd/anon_welcome.txt 这个档案中， 至于这个档案的内容你可以这样写 (这个档案一定要存在！否则会造成客户端无法联机成功喔！)：

```
[root@www ~]# vim /etc/vsftpd/anon_welcome.txt
欢迎光临本站所提供的 FTP 服务！
本站主要提供 Linux 操作系统相关档案以及 GNU 自由软件喔！
有问题请与站长联络！谢谢大家！
主要的目录为：

linux   提供 Linux 操作系统相关软件
gnu     提供 GNU 的自由软件
uploads 提供匿名的您上传数据
```

看到啰！主要写的数据都是针对一些公告事项就是了！

3. 客户端的测试：密码与欢迎讯息是重点！

同样的，我们使用 ftp 这个软件来给他测试一下吧！

```
[root@www ~]# ftp localhost
Connected to localhost (127.0.0.1).
220-欢迎光临本站所提供的 FTP 服务！   &lt;==底下这几行中文就是欢迎与提示讯息！
220-本站主要提供 Linux 操作系统相关档案以及 GNU 自由软件喔！
220-有问题请与站长联络！谢谢大家！
220-主要的目录为：
220-
220-linux    提供 Linux 操作系统相关软件
220-gnu      提供 GNU 的自由软件
220-uploads 提供匿名的您上传数据
220
Name (localhost:root): anonymous  &lt;==匿名账号名称是要背的！
230 Login successful.              &lt;==没有输入密码即可登入呢！
Remote system type is UNIX.
Using binary mode to transfer files.
ftp&gt; dir
227 Entering Passive Mode (127,0,0,1,196,17).
150 Here comes the directory listing.
drwxr-xr-x    2 0        0            4096 Aug 08 16:37 gnu
-rw-r--r--    1 0        0              17 Aug 08 14:18 index.html
drwxr-xr-x    2 0        0            4096 Aug 08 16:37 linux
drwxr-xr-x    2 0        0            4096 Jun 25 17:44 pub
226 Directory send OK.
ftp&gt; bye
221 Goodbye.
```

看到否？这次可就不需要输入任何密码了，因为是匿名登录嘛！而且，如果你以其他的账号来尝试登入时， 那么 vsftpd 会立刻响应仅开放匿名的讯息喔！(530 This FTP server is anonymous only.)

4. 让匿名者可上传/下载自己的资料 (权限开放最大)

在上列的数据当中，实际上匿名用户仅可进行下载的动作而已。如果你还想让匿名者可以上传档案或者是建立目录的话， 那你还需要额外增加一些设定才行：

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 新增底下这几行啊！
write_enable=YES
anon_other_write_enable=YES
anon_mkdir_write_enable=YES
anon_upload_enable=YES

[root@www ~]# /etc/init.d/vsftpd restart
```

如果你设定上面四项参数，则会允许匿名者拥有完整的建立、删除、修改档案与目录的权限。 不过，实际要生效还需要 Linux 的文件系统权限正确才行！ 我们知道匿名者取得的身份是 ftp ，所以如果想让匿名者上传数据到 /var/ftp/uploads/ 中，则需要这样做：

```
[root@www ~]# mkdir /var/ftp/uploads
[root@www ~]# chown ftp /var/ftp/uploads
```

然后你以匿名者身份登入后，就会发现匿名者的根目录多了一个 /upload 的目录存在了，并且你可以在该目录中上传档案/目录喔！ 如此一来系统的权限大开！很要命喔！所以，请仔细的控制好你的上传目录才行！

不过，在实际测试当中，却发现还是没办法上传呢！怎么回事啊？如果你有去看一下 /var/log/messages 的话，那就会发现啦！ 又是 SELinux 这家伙呢！怎么办？就透过『 sealert -l ... 』在 /var/log/messages 里面观察到的指令丢进去， 立刻就知道解决方案啦！解决方案就是放行 SELinux 的匿名 FTP 规则如下：

```
[root@www ~]# setsebool -P allow_ftpd_anon_write=1
[root@www ~]# setsebool -P allow_ftpd_full_access=1
```

然后你再测试一下用 anonymous 登入，到 /uploads 去上传个档案吧！就会知道能不能成功哩！

- 让匿名者仅具有上传权限，不可下载匿名者上传的东西

一般来说，用户上传的数据在管理员尚未查阅过是否合乎版权等相关事宜前，是不应该让其他人下载的！ 然而前一小节的设定当中，用户上传的资料是可以被其他人所浏览与下载的！ 如此一来实在是很危险！所以如果你要设定 /var/ftp/uploads/ 内透过匿名者上传的数据中，仅能上传不能被下载时，那么被上传的数据的权限就得要修改一下才行！ 请将前一小节所设定的四个参数简化成为：

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 将这几行给他改一改先！记得要拿掉 anon_other_write_enable=YES
write_enable=YES
anon_mkdir_write_enable=YES
anon_upload_enable=YES
chown_uploads=YES        &lt;==新增的设定值在此！
chown_username=daemon

[root@www ~]# /etc/init.d/vsftpd restart
```

当然啦，那个 /var/ftp/uploads/ 还是需要可以被 ftp 这个使用者写入才行！如此一来被上传的档案将会被修改档案拥有者成为 daemon 这个使用者，而 ftp (匿名者取得的身份) 是无法读取 daemon 的数据的，所以也就无法被下载啰！ ^_^

例题：在上述的设定后，我尝试以 anonymous 登入并且上传一个大档案到 /uploads/ 目录下。由于网络的问题，这个档案传到一半就断线。 下在我重新上传时，却告知这个档案无法覆写！该如何是好？答：为什么会无法覆写呢？因为这个档案在你脱机后，档案的拥有者就被改为 daemon 了！因为这个档案不属于 ftp 这个用户了， 因此我们无法进行覆写或删除的动作。此时，你只能更改本地端档案的档名再次的上传，重新从头一直上传啰！

- 被动式联机埠口的限制

FTP 的联机分为主动式与被动式，主动式联机比较好处理，因为都是透过服务器的 port 20 对外主动联机， 所以防火墙的处理比较简单。被动式联机就比较麻烦～因为预设 FTP 服务器会随机取几个没有在使用当中的埠口来建立被动式联机，那防火墙的设定就麻烦啦！

没关系，我们可以透过指定几个固定范围内的埠口来作为 FTP 的被动式数据连接之用即可，这样我们就能够预先知道 FTP 数据链路的埠口啦！举例来说，我们假设被动式连接的埠口为 65400 到 65410 这几个埠口时，可以这样设定：

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 增加底下这几行即可啊！
pasv_min_port=65400
pasv_max_port=65410

[root@www ~]# /etc/init.d/vsftpd restart
```

匿名用户的设定大致上这样就能符合你的需求啰！其他的设定就自己看着办吧！ ^_^

21.2.8 防火墙设定

防火墙设定有什么难的？将第九章里面的 script 拿出来修改即可啊！不过，如同前言谈到的，FTP 使用两个埠口，加上常有随机启用的数据流埠口，以及被动式联机的服务器埠口等， 所以，你可能得要进行：

- 加入 iptables 的 ip_nat_ftp, ip_conntrack_ftp 两个模块
- 开放 port 21 给因特网使用
- 开放前一小节提到的 port 65400~65410 埠口给 Internet 联机用

要修改的地方不少，那就让我们来一步一脚印吧！

```
# 1\. 加入模块：虽然 iptables.rule 已加入模块，不过系统档案还是修改一下好了：
[root@www ~]# vim /etc/sysconfig/iptables-config
IPTABLES_MODULES="ip_nat_ftp ip_conntrack_ftp"
# 加入模块即可！两个模块中间有空格键隔开！然后重新启动 iptables 服务啰！

[root@www ~]# /etc/init.d/iptables restart

# 2\. 修改 iptables.rule 的脚本如下：
[root@www ~]# vim /usr/local/virus/iptables/iptables.rule
iptables -A INPUT -p TCP -i $EXTIF --dport  21  --sport 1024:65534 -j ACCEPT
# 找到上面这一行，并将前面的批注拿掉即可！并且新增底下这一行喔！
iptables -A INPUT -p TCP -i $EXTIF --dport 65400:65410 --sport 1024:65534 -j ACCEPT

[root@www ~]# /usr/local/virus/iptables/iptables.rule
```

这样就好了！同时兼顾主动式与被动式的联机！并且加入所需要的 FTP 模块啰！

21.2.9 常见问题与解决之道

底下说明几个常见的问题与解决之道吧！

- 如果在 Client 端上面发现无法联机成功，请检查：

1. iptables 防火墙的规则当中，是否开放了 client 端的 port 21 登入？
2. 在 /etc/hosts.deny 当中，是否将 client 的登入权限挡住了？
3. 在 /etc/xinetd.d/vsftpd 当中，是否设定错误，导致 client 的登入权限被取消了？

- 如果 Client 已经连上 vsftpd 服务器，但是却显示『 XXX file can't be opend 』的字样，请检查：

   1. 最主要的原因还是在于在 vsftpd.conf 当中设定了检查某个档案，但是你却没有将该档案设定起来， 所以，请检查 vsftpd.conf 里面所有设定的档案档名，使用 touch 这个指令将该档案建立起来即可！

- 如果 Client 已经连上 vsftpd 服务器，却无法使用某个账号登入，请检查：

   1. 在 vsftpd.conf 里面是否设定了使用 pam 模块来检验账号，以及利用 userlist_file 来管理账号？
   2. 请检查 /etc/vsftpd/ftpusers 以及 /etc/vsftpd/user_list 档案内是否将该账号写入了？

- 如果 Client 无法上传档案，该如何是好？

   1. 最可能发生的原因就是在 vsftpd.conf 里面忘记加上这个设定『write_enable=YES』这个设定，请加入；
   2. 是否所要上传的目录『权限』不对，请以 chmod 或 chown 来修订；
   3. 是否 anonymous 的设定里面忘记加上了底下三个参数：
      - anon_other_write_enable=YES
      - anon_mkdir_write_enable=YES
      - anon_upload_enable=YES
   4. 是否因为设定了 email 抵挡机制，又将 email address 写入该档案中了！？请检查！
   5. 是否设定了不许 ASCII 格式传送，但 Client 端却以 ASCII 传送呢？请在 client 端以 binary 格式来传送档案！
   6. 检查一下 /var/log/messages ，是否被 SELinux 所抵挡住了呢？

上面是蛮常发现的错误，如果还是无法解决你的问题，请你务必分析一下这两个档案：/var/log/vsftpd.log 与 /var/log/messages ，里面有相当多的重要资料，可以提供给你进行除错喔！不过 /var/log/vsftpd.log 却预设不会出现！ 只有 /var/log/xferlog 而已。如果你想要加入 /var/log/vsftpd.log 的支持，可以这样做：

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
dual_log_enable=YES
vsftpd_log_file=/var/log/vsftpd.log
# 加入这两个设定值即可呦！

[root@www ~]# /etc/init.d/vsftpd restart
```

这样未来有新联机或者是错误时，就会额外写一份 /var/log/vsftpd.log 去喔！

21.3 客户端的图形接口 FTP 联机软件

客户端的联机软件主要有文字接口的 ftp 及 lftp 这两支指令，详细的使用方式请参考第五章常用网络指令的说明。至于 Linux 底下的图形接口软件，可以参考 gftp 这支程序喔！图形接口的啦！很简单啊！那 Windows 底下有没有相对应的 FTP 客户端软件？

21.3.1 Filezilla

上述的软件都是自由软件啊，那么 Windows 操作系统有没有自由软件啊？有的，你可以使用 filezilla 这个好东西！这个玩意儿的详细说明与下载点可以在底下的连结找到：

- 说明网站：http://filezilla.sourceforge.net/
- 下载网站：http://sourceforge.net/project/showfiles.php?group_id=21558

目前 (2011/06) 最新的稳定版本是 3.5.x 版，所以底下鸟哥就以这个版本来跟大家说明。为什么要选择 Filezilla 呢？除了他是自由软件之外，这家伙竟然可以连结到 SSH 的 sftp 呢！真是很不错的一个家伙啊！^^！另外要注意的是，底下鸟哥是以 *Windows* 版本来说明的，不要拿来在 *X window* 上面安装喔！^^ (请下载 Filezilla client 不是 server 喔！)

因为这个程序是给 Windows 安装用的，所以安装的过程就是...(下一步)^n 就好了！并且这个程序支持多国语系， 所以你可以选择繁体中文呢！实在是很棒！安装完毕之后，请你执行他，就会出现如下的画面了：

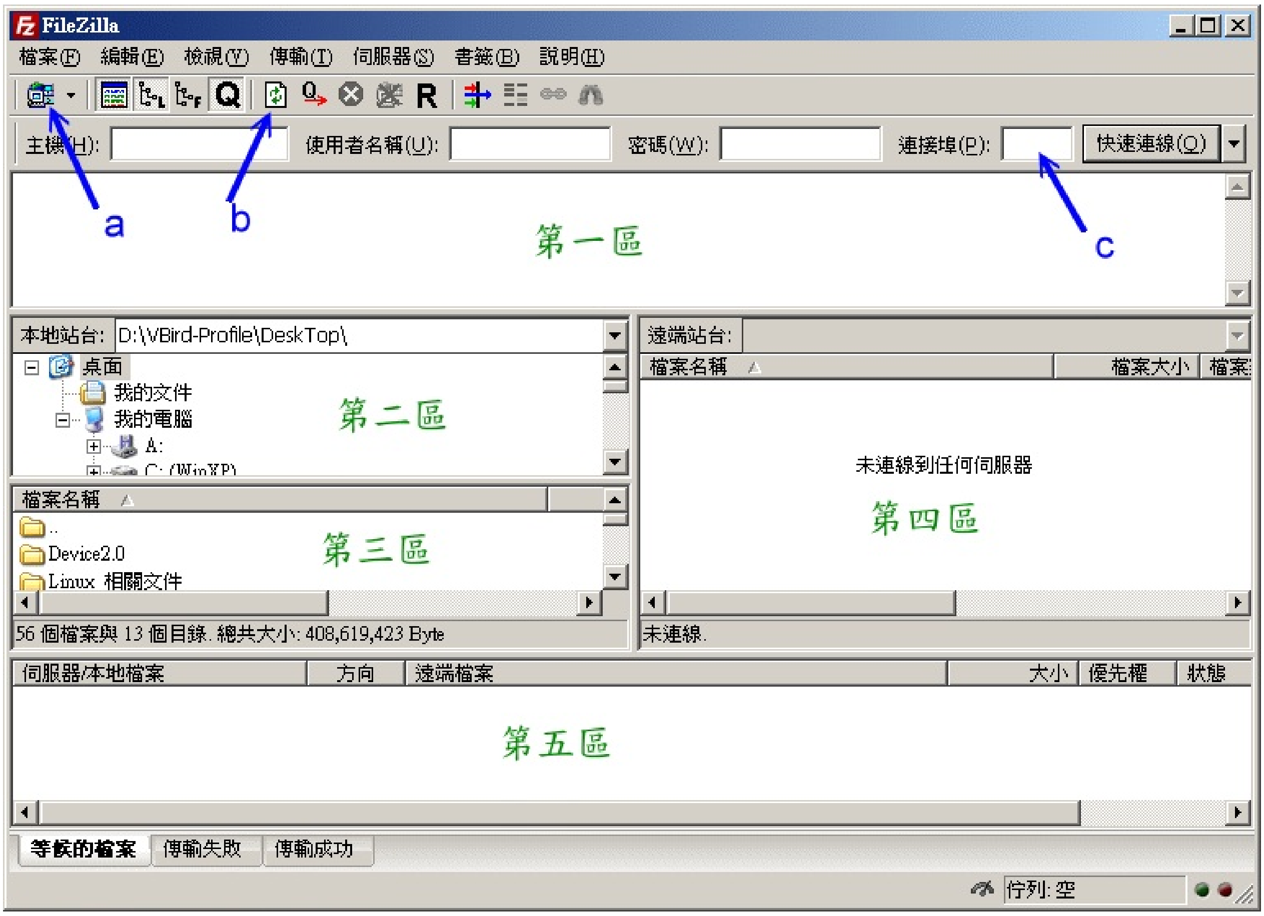


图 21.3-1、Filezilla 的操作接口示意图

上图的 第一、二到五区的内容所代表的资料是：

1. 第一区：代表 FTP 服务器的输出信息，例如欢迎讯息等信息；
2. 第二区：代表本机的文件系统目录，与第三区有关；
3. 第三区：代表第二区所选择的磁盘内容为何；
4. 第四区：代表远程 FTP 服务器的目录与档案；
5. 第五区：代表传输时的队列信息 (等待传送的数据)

而另外图中的 a, b, c 则代表的是：

1. 站台管理员，你可以将一些常用的 FTP 服务器的 IP 与用户信息记录在此；
2. 更新，如果你的资料有更新，可使用这个按钮来同步 filezilla 的屏幕显示；
3. 主机地址、用户、密码与端口这四个玩意儿可以实时联机，不记录信息。

好，接下来我们连接到 FTP 服务器上面去，所以你可按下图 21.3-1 的 a 部分，会出现如下画面：



图 21.3-2、Filezilla 的 FTP 站台管理员使用示意图

上图的箭头与相关的内容是这样的：

1. 先按下『新增站台』的按钮，然后在箭头 2 的地方就会出现可输入名称的方框；
2. 在该方框当中随便填写一个你容易记录的名字，只要与真正的网站有点关连即可；
3. 接下来看到右边有一般设定，在一般设定里面几个项目很重要的：
    - 主机：在这个方框中填写主机的 IP，端口如果不是标准的 port 21 才填写其他埠口。
    - 协定：主要有 (1)FTP 及 (2)SFTP (SSHD 所提供)，我们这里选 FTP
    - 加密：是否有网络加密，新的协议中，FTP 可以加上 TLS 的 FTPS 喔！预设为明码
    - 登入型式：因为需要账号密码，选择『一般』即可，然后底下就是输入使用者、账号即可。

基本上这样设定完就能够连上主机了，不过，如果你还想要更详细的规范数据连接的方式 (主动式与被动式) 以及其他数据时， 可以按下的『传输设定』按钮，就会出现如下画面了：

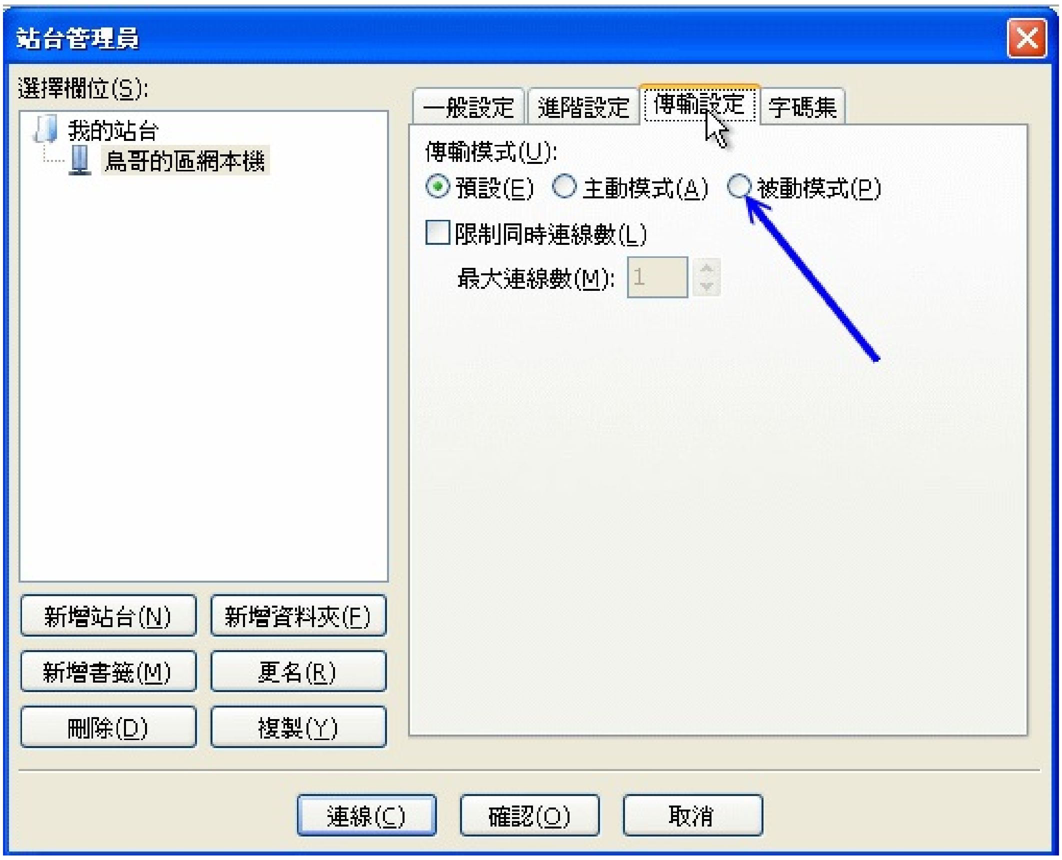


图 21.3-3、Filezilla 站台管理员内的传输设定

在这个画面当中你可以选择是否使用被动式传输机制，还可以调整最大联机数呢！为什么要自我限制呢？因为 Filezilla 会主动的重复建立多条联机来快速下载，但如果 vsftpd.conf 有限制 max_per_ip 的话，某些下载会被拒绝的！因此，这个时候在此设定为 1 就显的很重要～随时只有一支联机建立，就不会有重复登入的问题！ 最后请按下图 21.3-2 画面中的『联机』吧！

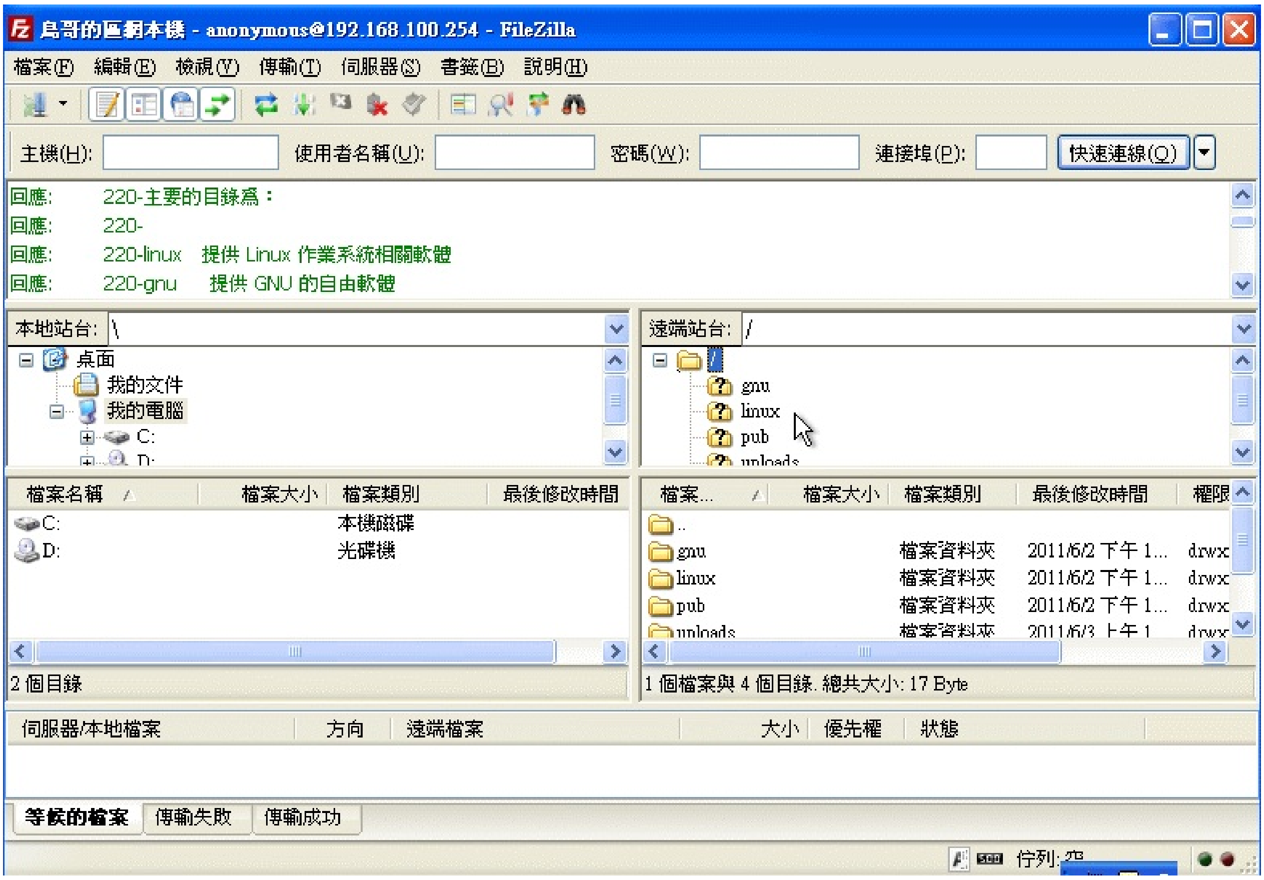


图 21.3-4、Filezilla 联机成功示意图

更多的用法就请你自行研究啰！

21.3.2 透过浏览器取得 FTP 联机

我们在 第二十章 WWW 服务器当中曾经谈过浏览器所支持的协议，其中一个就是 ftp 这个协定啰！这个协议的处理方式可以在网址列的地方这样输入的：

- ftp://username@your_ip

要记得，如果你没有输入那个 username@ 的字样时，系统默认会以匿名登录来处理这次的联机。因此如果你想要使用实体用户联机时， 就在在 IP 或主机名之前填写你的账号。举例来说，鸟哥的 FTP 服务器 (192.168.100.254) 若有 dmtsai 这个使用者， 那我启动浏览器后，可以这样做：

- ftp://dmtsai@192.168.100.254

然后在出现的对话窗口当中输入 dmtsai 的密码，就能够使用浏览器来管理我在 FTP 服务器内的文件系统啰！是否很容易啊 甚至，你连密码都想要写上网址列，那就更厉害啦！

- ftp://dmtsai:yourpassword@192.168.100.254

21.4 让 vsftpd 增加 SSL 的加密功能

既然 http 都有 https 了，那么使用明码传输的 ftp 有没有加密的 ftps 呢？嘿嘿！说的好！有的啦～既然都有 openssl 这个加密函式库， 我们当然能够使用类似的机制来处理 FTP 啰！但前提之下是你的 vsftpd 有支持 SSL 函式库才行！此外，我们也必须要建立 SSL 的凭证档给 vsftpd 使用，这样才能够进行加密嘛！了解乎！接下来，就让我们一步一步的进行 ftps 的服务器建置吧！

- 1. 检查 vsftpd 有无支持 ssl 模块：

如果你的 vsftpd 当初编译的时候没有支持 SSL 模块，那么你就得只好自己重新编译一个 vsftpd 的软件了！我们的 CentOS 有支持吗？ 赶紧来瞧瞧：

```
[root@www ~]# ldd $(which vsftpd) &#124; grep ssl
        libssl.so.10 =&gt; /usr/lib64/libssl.so.10 (0x00007f0587879000)
```

如果有出现 libssl.so 的字样，就是有支持！这样才能够继续下一步呦！

- 2. 建立专门给 vsftpd 使用的凭证数据：

CentOS 给我们一个建立凭证的地方，那就是 /etc/pki/tls/certs/ 这个目录！详细的说明我们在 20.5.2 里面谈过咯，所以这里只介绍怎么做：

```
[root@www ~]# cd /etc/pki/tls/certs
[root@www certs]# make vsftpd.pem
----- ....(前面省略)....
Country Name (2 letter code) [XX]:TW
State or Province Name (full name) []:Taiwan
Locality Name (eg, city) [Default City]:Tainan
Organization Name (eg, company) [Default Company Ltd]:KSU
Organizational Unit Name (eg, section) []:DIC
Common Name (eg, your name or your server's hostname) []:www.centos.vbird
Email Address []:root@www.centos.vbird

[root@www certs]# cp -a vsftpd.pem /etc/vsftpd/
[root@www certs]# ll /etc/vsftpd/vsftpd.pem
-rw-------. 1 root root 3116 2011-08-08 16:52 /etc/vsftpd/vsftpd.pem
# 要注意一下权限喔！
```

- 3. 修改 vsftpd.conf 的配置文件，假定有实体、匿名账号：

在前面 21.2 里面大多是单纯匿名或单纯实体帐户，这里我们将实体账号透过 SSL 联机，但匿名者使用明码传输！ 两者同时提供给客户端使用啦！FTP 的设定项目主要是这样：

- 提供实体账号登入，实体账号可上传数据，且 umask 为 002
- 实体账号默认为 chroot 的情况，且全部实体账号可用带宽为 1Mbytes/second
- 实体账号的登入与数据传输均需透过 SSL 加密功能传送；
- 提供匿名登录，匿名者仅能下载，不能上传，且使用明码传输 (不透过 SSL)

此时，整体的设定值会有点像这样：

```
[root@www ~]# vim /etc/vsftpd/vsftpd.conf
# 实体账号的一般设定项目：
local_enable=YES
write_enable=YES
local_umask=002
chroot_local_user=YES
chroot_list_enable=YES
chroot_list_file=/etc/vsftpd/chroot_list
local_max_rate=10000000

# 匿名者的一般设定：
anonymous_enable=YES
no_anon_password=YES
anon_max_rate=1000000
data_connection_timeout=60
idle_session_timeout=600

# 针对 SSL 所加入的特别参数！每个项目都很重要！
ssl_enable=YES              &lt;==启动 SSL 的支持
allow_anon_ssl=NO           &lt;==但是不允许匿名者使用 SSL 喔！
force_local_data_ssl=YES    &lt;==强制实体用户数据传输加密
force_local_logins_ssl=YES  &lt;==同上，但连登入时的帐密也加密
ssl_tlsv1=YES               &lt;==支持 TLS 方式即可，底下不用启动
ssl_sslv2=NO
ssl_sslv3=NO
rsa_cert_file=/etc/vsftpd/vsftpd.pem &lt;==预设 RSA 加密的凭证档案所在

# 一般服务器系统设定的项目：
max_clients=50
max_per_ip=5
use_localtime=YES
dirmessage_enable=YES
xferlog_enable=YES
connect_from_port_20=YES
xferlog_std_format=YES
listen=YES
pam_service_name=vsftpd
tcp_wrappers=YES
banner_file=/etc/vsftpd/welcome.txt
dual_log_enable=YES
vsftpd_log_file=/var/log/vsftpd.log
pasv_min_port=65400
pasv_max_port=65410

[root@www ~]# /etc/init.d/vsftpd restart
```

- 4. 联机测试看看！使用 Filezilla 联机测试：

接下来我们利用 filezilla 来说明一下，如何透过 SSL/TLS 功能来进行联机加密。很简单，只要在站台管理员的地方选择：

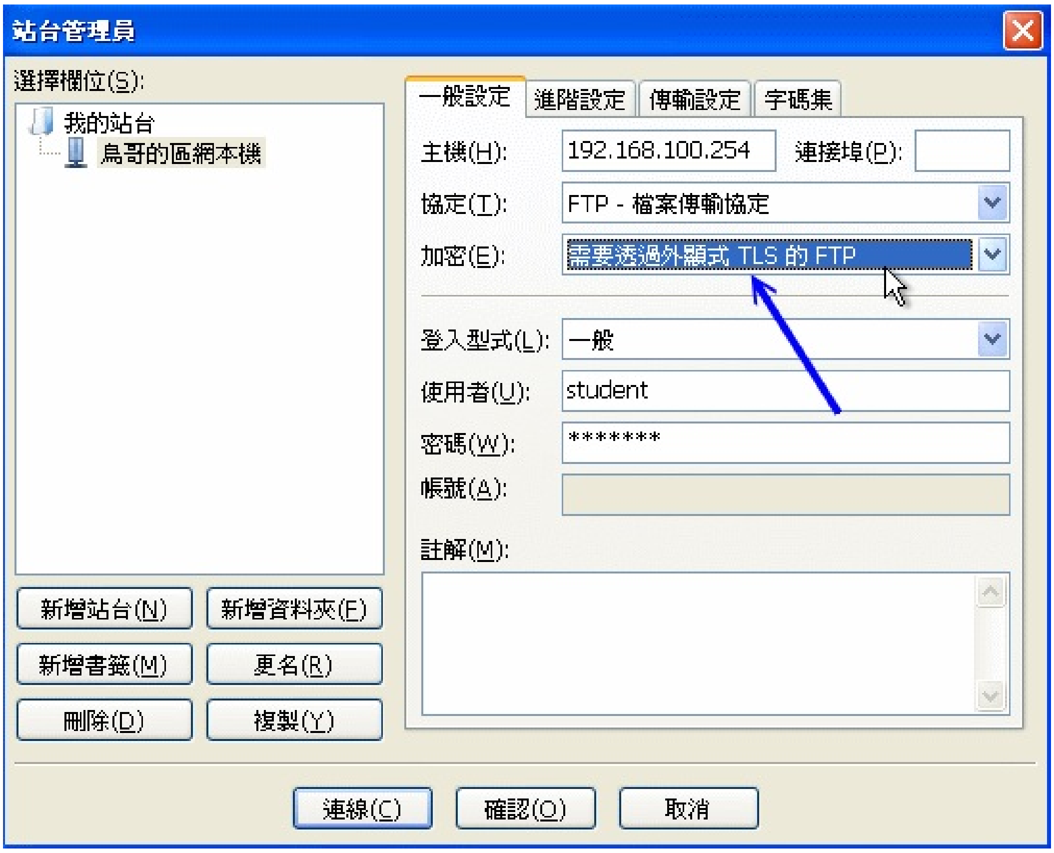


图 21.4-1、透过 Filezilla 联机到 SSL/TLS 支持的 FTP 方式

如上图所示，重点在箭头所指的地方，需要透过 TLS 的加密方式才行！然后，鸟哥尝试使用 student 这个一般账号登入系统， 联机的时候，应该会出现如下的图示才对：



图 21.4-2、透过 Filezilla 是否接受凭证呢？

如果一切都没有问题，那么你可以点选上图那个『总是信任』的项目，如此一来，未来联机到这个地方就不会再次要你确认凭证啦！ 很简单的解决了 FTP 联机加密的问题啰！^_^

例题：想一想，既然有了 SFTP 可以进行加密的 FTP 传输，那为何需要 ftps 呢？答：因为既然要开放 SFTP 的话，就得要同时放行 sshd 亦即是 ssh 的联机，如此一来，你的 port 22 很可能会常常被侦测～若是 openssl, openssh 出问题，恐怕你的系统就会被绑架。如果你的 FTP 真的有必要存在，那么透过 ftps 以及利用 vsftpd 这个较为安全的服务器软件来架设， 理论上，是要比 sftp 来的安全些～至少对 Internet 放行 ftps 还不会觉得很可怕...

21.5 重点回顾

- FTP 是文件传输协议 (File Transfer Protocol) 的简写，主要的功能是进行服务器与客户端的档案管理、传输等事项；
- FTP 的服务器软件非常多，例如 Wu FTP, Proftpd, vsftpd 等等，各种 FTP 服务器软件的发展理念并不相同， 所以选择时请依照你的需求来决定所需要的软件；
- FTP 使用的是明码传输，而过去一些 FTP 服务器软件也曾被发现安全漏洞，因此设定前请确定该软件已是最新版本，避免安全议题的衍生；
- 由于 FTP 是明码传输，其实可以使用 SSH 提供的 sftp 来取代 FTP ；
- 大多数的 FTP 服务器软件都提供 chroot 的功能，将实体用户限制在他的家目录内；
- FTP 这个 daemon 所开启的正规埠口为 20 与 21 ，其中 21 为命令通道， 20 为主动联机的数据传输信道；
- FTP 的数据传输方式主要分为主动与被动(Passive, PASV)，如果是主动的话，则 ftp-data 在服务器端主动以 port 20 连接到客户端，否则需开放被动式监听的埠口等待客户端来连接；
- 在 NAT 主机内的客户端 FTP 软件联机时可能发生困扰，这可以透过 iptables 的 nat 模块或利用被动式联机来克服；
- 一般来说， FTP 上面共有三个群组，分别是实体用户 、访客与匿名登录者(real, guest, anonymous)；
- 可以藉由修改 /etc/passwd 里面的 Shell 字段，来让使用者仅能使用 FTP 而无法登入主机；
- FTP 的指令、与用户活动所造成的登录档是放置在 /var/log/xferlog 里面；
- vsftpd 为专注在安全议题上而发展的一套 FTP 服务器软件，他的配置文件在 /etc/vsftpd/vsftpd.conf

21.6 本章习题

- FTP 在建立联机以及数据传输时，会建立哪些联机？需建立两种联机，分别是命令信道与数据传输信道。在主动式联机上为 port 21(ftp) 与 port 20(ftp-data)。
- FTP 主动式与被动式联机有何不同？主动式联机的时候，命令联机是由 client 端主动连接到服务器端，但是 ftp-data 则是由服务器端主动的联机到 client 端。至于被动式联机的时候，则不论 command 还是 ftp-data 的联机，服务器端都是监听客户端的要求的！
- 有哪些动作可以让你的 FTP 主机更为安全 (secure) ？
  - 随时更新服务器软件到最新版本；
  - 让 guest 与 anonymous 的家目录限制在固定的目录中(chroot 或是 restricted)；
  - 拒绝 root 的登入或者其他系统账号的登入；
  - 拒绝大部分的 upload 行为！
- 我们知道 ftp 会启用两个 ports ，请问这两个 port 在哪里规范的 (以 vsftpd 为例)？而且，一般正规的 port 是几号？若为 stand alone 时，都是由 vsftpd.conf 规范，命令通道为 listen_port=21 规范，数据连接为 connect_from_port_20=YES 及pasv_max_port=0, pasv_max_port=0 所规范。 若是 super daemon 所管理时，命令信道则由 /etc/services 所规范了。
- 那几个档案可以用来抵挡类似 root 这种系统账号的登入 FTP？/etc/vsftpd/ftpusers /etc/vsftpd/user_list
- 在 FTP 的 server 与 client 端进行数据传输时，有哪两种模式？为何这两种模式影响数据的传输很重要？数据的传输有 ASCII 与 Binary 两种方式，在进行 ascii 传送方式时，被传送的档案将会以文本模式来进行传送的行为， 因此，档案的属性会被修改过，可能造成执行档最后却无法执行等的问题！一般来说，ASCII 通常仅用在文本文件与一些原始码档案的传送。
- 我的主机明明时区设定没有问题，但为何登入 vsftpd 这个 FTP 服务时，时间就是少八小时？该如何解决？肯定是时区方面出了问题，应该就是 vsftpd.conf 里面少了『 use_localtime=YES 』这个参数了。

21.7 参考数据与延伸阅读

- vsftpd 官方网站：http://vsftpd.beasts.org/
- man 5 vsftpd.conf
- Filezilla 官方网站：http://filezilla.sourceforge.net/
- vsftpd + ssl 功能：http://wiki.vpslink.com/Configuring*vsftpd_for_secure_connections*%28TLS/SSL/SFTP%29
- http://beginlinux.com/blog/2009/01/secure-ftp-with-ssl-on-centos/

2003/09/03：首次完成 2003/09/04：加入 FTP 服务器软件的选择建议 2006/12/19：将旧的文章移动到此处，并请自行参考 wu-ftp, proftpd 等服务！ 2006/12/20：将分散在各处的 FTP 原则说明完毕，也更新完毕啰～疲劳～ 2011/05/28：将旧的基于 CentOS 4.x 的文章移动到此处 2011/06/04：加入了 ftps 的 SSL 联机加密机制！ 2011/08/08：将基于 CentOS 5.x 的版本移动到 此处

第二十二章、邮件服务器： Postfix

最近更新日期：2011/08/10

在这个邮件服务器的架设中，我们首先谈论 Mail 与 DNS 的重要相关性，然后依序介绍 Mail Server 的相关名词，以及 Mail Server 的运作基本流程与协议，也会谈到相关的 Relay 与邮件认证机制等项目， 这些项目对于未来邮件服务器的管理与设定是重要的，请不要忽略了这方面问题的讨论喔。 由于 Postfix 的配置文件内容较具有亲和性，因此我们单纯介绍了 Postfix 不再介绍 sendmail 了。

22.1 邮件服务器的功能与运作原理

电子邮件是个啥玩意儿？它是利用网络传递一些信息给远程服务器的一种信息传递行为，虽然消息正文是很冷很硬的计算机文字， 确实比不上手写信件来的让人觉得温暖，不过，对于具有时效性的信息来说，电子邮件可是个不可多得的好帮手！ 但是，电子邮件系统蓬勃发展的现在却被某些少部分的特定人士所乱用，导致垃圾信件、色情广告信件等等的泛滥！ 眞是啊～伤脑筋～底下我们就先来谈一谈这个电子邮件相关的功能吧！

Tips: 时至今日，Google 与几个大型的网络公司都有提供免费或者是付费的邮件服务器，其中，免费的电子邮件账号甚至已经提供高达数个 GB 的邮件储存量！对于一般用户来说眞是非常够用了！因此，除非必要，现在我们都『不建议您架设 mail server』的！ 因为玩过邮件主机的朋友都很清楚，在现在的环境当中想要搞定 Mail server 是很难的一件事情， 除了目前网络社会的广告信、垃圾信、病毒信实在是多的不象话，所以各主要的 ISP 对于邮件控管上面越来越严格， 而且基本功当中的 mail vs. DNS 相关性又太高！很难理解～



22.1.1 电子邮件的功能与问题

在目前的社会当中，没有电子邮件 (e-mail) 似乎是蛮奇怪的一件事！可以说，现在 e-mail 已经成为一个很普遍的人与人之间的沟通管道了， 电子邮件可以很快速的帮你将文件或讯息传送到地球上的任何一个有网络存在的角落，当然， 你也可以在任何有网络的地方，连上 Internet 去收取你的信件！

不过，遗憾的是，只要是有人类的地方，就会有很多你意想不到的事情会出现了，当然 e-mail 也不例外，怎么说呢？ 我们来慢慢的分析一下电子邮件产生的一些问题吧：

- 夹带病毒的电子邮件问题： 你可以常常听到电子邮件可能夹带病毒对吧！没错，利用电子邮件以及人们对于电子邮件的漫不经心的态度， 使得以电子邮件为媒介的计算机病毒更容易『深入人群』当中呐！
- 怪客透过邮件程序入侵： 只要在 Internet 上面跑的数据就没有绝对保密的！你可以轻易的使用怪客软件 (Cracker) 就可以取得使用者在利用 e-mail 传送过程当中所输入的账号与密码，若经过分析之后，还可能破解对方的邮件主机～哇！眞是乱可怕一把的！

- 广告信与垃圾信等： 这个可说是目前各大 ISP 心中永远的痛～这些垃圾信件可以占掉很多那少的可怜的带宽， 使得正常用户连接速度与质量下降，更可能造成网络的停顿～当然，常常收到垃圾信件的你，大概也不好过吧！
- 主机被大量不明信件塞爆： 万一你没有将邮件服务器设定好，嘿嘿！送信者可以藉由你主机收信的功能，发送大量的信件， 让你『一次收个够！』灌爆你的服务器硬盘，想要不当机都粉难～
- 眞实社会的讨厌情事： 『黑函』！听到会不会很害怕？当然很害怕啦！偏偏使用 e-mail 就可以作很多的坏事～这眞是太不道德了～
- 不实的信件内容： 只要注意到消基会的讯息就可以知道啦，不明来源的电子邮件说的内容，不要轻易的相信！ 因为很多可是以讹传讹，结果，大家都被耍了的～例如，你的朋友收到一封信，认为『哇！这是大事情』， 所以在没有求证的情况下，将信『转寄』给你看，嘿！你的朋友寄给你的，当然要相信他啦！ 立刻再转寄，如此一再地循环，嘿嘿！这个错误内容的讯息马上就让大家知道， 更可怕的是『还会让大家接受～』所以，看到任何讯息时，请千万要记得求证一下呐！

可怕吧！电子邮件会衍生出这么多的问题说～另外，这个 email 服务器的设定与管理眞的是网管人员心中永远的痛！ 为什么呢？因为人都是想要越便利越简单越好，但越便利越不管制的邮件服务器就越容易被攻击或遭利用！ 反过来说，如果你针对邮件服务器管得太严厉，那就不太人性化，相信至少您的主管可能就不太满意，怎么办？

呵呵！没错啦！邮件服务器就是这么回事，让人又爱又怕的一个玩意儿，搞定他，恭喜您啊一切顺利圆满！ 搞不定他，服务器被当成垃圾信件转运站事小，丢掉工作那可是『兹事体大』呦！就因为他是这么重要又难以搞定， 所以我们可得好好的学学他呐！

22.1.2 Mail server 与 DNS 之间的关系

既然要使用 e-mail ，当然就需要邮件服务器啰 (Mail Server)！不然你的信要怎样寄出去呢？ 事实上， mail server 的原理说难不难，但是说简单吗～似乎又有点难以理解～，所以， 底下我们要来谈一谈他的原理部分，然后再针对服务器的设定来进行说明咯！我们首先要讲的就是『 Mail server 系统与 DNS 系统有什么关连性？』 这个部分新手最容易被搞混哩，是否要架设 mail server 就『宿命』的一定得架设 DNS server 在你的主机上面吗？

- Mail server 与合法的主机名

事实上目前已经没有人会使用 IP 来寄信了，我们通常接收到的 email 都是使用『账号@主机名』的方式来处理的， 所以说，你的邮件服务器『就一定要有一个合法注册过的主机名』才可以。为什么呢？ 因为网络恶意使用与垃圾邮件泛滥的种种因素，导致我们不允许直接利用主机的 IP 来寄信了，否则每部有 IP 的主机都能寄信... 因此，你想要架设 mail server 就『必需』要有合法的主机名啰。

OK！既然我只要一个合法的主机名即可，那么表示我不需要架设一部 DNS 主机啰？ 是的，你可以这样认为！只要你拥有合法的主机名，亦即在 DNS 的查询系统当中你的主机名拥有一个 A 的标志， 理论上你的 mail server 就可以架设成功。只不过由于目前因特网上面的广告信、 垃圾信与病毒信等占用了太多的带宽，导致整个网络社会花费过多的成本在消耗这些垃圾资料。 所以为了杜绝可恶的垃圾信件，目前的大型网络供货商 (ISP) 都会针对不明来源的邮件加以限制， 这也就是说『想要架设一部简单可以运作的 mail server 越来越难了』。

- DNS 的反解也很重要！

对于一般的服务器来说，我们只要使用正解让客户端可以正确的找到我们服务器的 IP 即可架站，举例来说 WWW 服务器就是这样。不过，由于目前收信端的邮件服务器会针对邮件来源的 IP 进行反解，而如果你的网络环境是由拨接取得非固定的 IP 时，该种 IP 在 ISP 方面通常会主动的以 xxx.dynamic.xxx 之类的主机名来管理，偏偏这样的主机名会被主要的大型邮件服务器 (例如 hotmail, yahoo 等) 视为垃圾信件， 所以你的邮件服务器所发出的信件将可能被丢弃，那可就伤脑筋了！

所以啊，如果你想要架设一部 Mail server 的话，请『务必』向您的上层 ISP 申请 IP 反解的对应， 不要再使用预设的反解主机名，否则很容易导致您的邮件服务器所发出的信件会在 Internet 上面流浪啊！

Tips: 其实你还是可以不用申请 IP 的反解，不过就得要利用所谓的 relayhost 或者是 smarthost 来处理邮件转递的问题， 这个部分又涉及到上层 ISP 的问题，挺复杂！我们会在后续作说明！



- 需要 DNS 的 MX 及 A 标志啊 (超重要的 MX)！

那么我们的邮件服务器系统到底是如何使用 DNS 的信息来进行邮件的传递的？还记得在十九章 DNS 里面谈到的 MX 这个标志吗？当时我们仅说过这个 MX 代表的是 Mail eXchanger，当一封邮件要传送出去时，邮件主机会先分析那封信的『目标主机的 DNS 』，先取得 MX 标志 (注意，MX 标志可能会有多部主机喔) 然后以最优先 MX 主机为准将信发送出去。看不懂吗？没关系，我们以底下这个 DNS 范例来说：

```
xyz.com.vbird   IN   MX 10 mail.xyz.com.vbird
xyz.com.vbird   IN   MX 20 mail2.xyz.com.vbird
xyz.com.vbird   IN   A     aaa.bbb.ccc.ddd
```

假如上述的 DNS 设定是正常的，那么：

- 当有一封信要传给 user@xyz.com.vbird 时，由于 MX 标志最低者优先，所以该封信会先传送到 mail.xyz.com.vbird 那部主机。

- 如果 mail.xyz.com.vbird 由于种种原因，导致无法收下该封信时，该封信将以次要 MX 主机来传送，那就是传送到 mail2.xyz.com.vbird 那部主机上头；
- 如果两部 MX 主机都无法负责的话，那么该封信会直接以 A 的标志，亦即直接传送到 aaa.bbb.ccc.ddd 那个 IP 上头去， 也就是 xyz.com.vbird 本身啦！

在这个过程当中，你必需要注意到：mail.xyz.com.vbird 及 mail2.xyz.com.vbird 必需要是可以帮 xyz.com.vbird 转信的主机才行，也就是说，那两部主机通常是你公司的最上游的邮件主机， 并不是你随意填写的！那两部主机还需要针对你的 xyz.com.vbird 来设定『邮件转递』才行！ 否则你的信会被踢掉的。

由于现在的很多邮件服务器会去搜寻 MX 这个标志来判断目标邮件服务器是否为合法，所以你要架设 Mail server 虽然不必自行设定 DNS 服务器，不过你最好要申请一个 MX 的标志才行。此外，MX 标志一定要设定正确，否则你的信件将可能会直接被 MX 服务器踢掉。为了要设定 MX 但是我们没有上层邮件服务器时，所以你可以指定 MX 为自己，利用自己当 MX 服务器即可。

那么你或许会想，这个 MX 有啥好处啊？一般来说，如果目标主机挂点时，你的邮件通常会直接退还给原发信者， 但如果有 MX 主机时，这部 MX 主机会先将该封信放在他的队列 (queue) 当中，等到你的目标主机重新提供邮件服务后， MX 主机会将你的信件传送给目标主机，如此一来你的信件就比较不会遗失啊！这样说，您可以了解吧！ ^_^

- Email 的地址写法

刚刚上头说过 email 通常是『账号@主机名』的方式来处理，举例来说鸟哥的 www.centos.vbird 主机上面有个 dmtsai 的使用者，则我的 email 将会成为：『dmtsai@www.centos.vbird』，当有人要寄信给我时， 他会分析 @ 后面的主机名，亦即 www.centos.vbird 的 MX/A 标志等等，然后再透过刚刚说明的流程来传出信件。 而当我的 www.centos.vbird. 收到这封信时，他会将信放到 dmtsai 的信箱当中啦！底下我们就来谈一谈这个流程吧！

22.1.3 邮件传输所需要的组件 (MTA, MUA, MDA) 以及相关协议

在开始介绍邮件的传送过程之前，我们先来想一想，你是如何寄出电子邮件的？假设你要寄信给一个使用者， 他的电子邮件是『a_user@gmail.com』好了，也就是说，你要寄一封信到 gmail.com 这个主机上的意思。 那你的桌面计算机 (举例来说， Windows 系统) 是否能够将这封信『直接』透过网络送给 gmail.com 那个主机上？ 当然不行啦！你得要设定帮你转信的邮件服务器才行！也就是说，你必需要先向某一部邮件服务器注册， 以取得一个合法的电子邮件权限后，才能够发送邮件出去的。

所以说，你要寄出一封信件时是需要很多接口的帮忙的，底下列出一个简单的图示来说明：

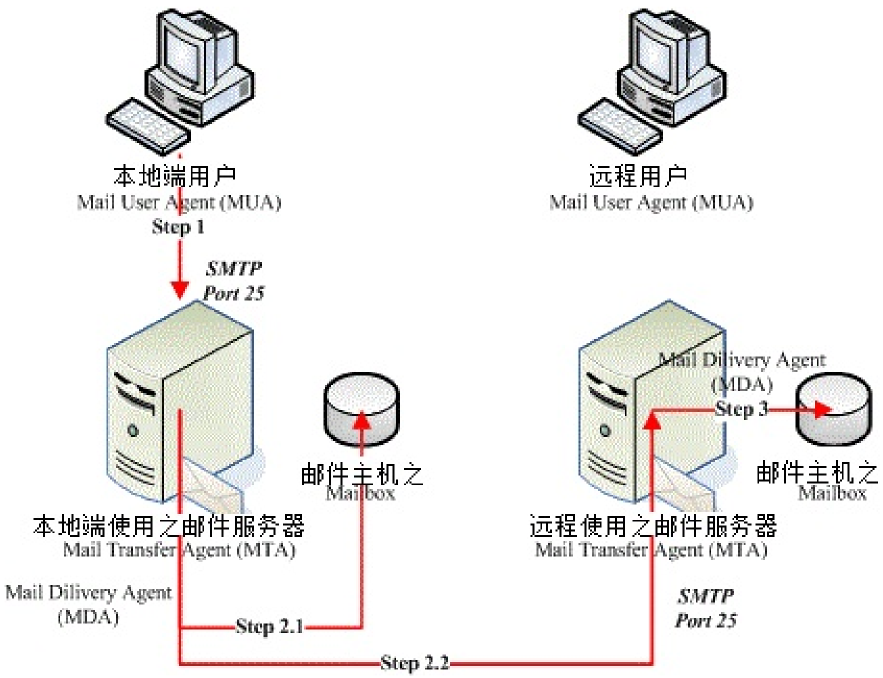


图 22.1-1、电子邮件的『传送』过程示意图

我们先来解释一些专有名词吧！然后再来说明传送的流程：

1. MUA (Mail User Agent)：

   顾名思义 MUA 就是『邮件使用者代理人』的意思，因为除非你可以直接利用类似 telnet 之类的软件登入邮件服务器来主动发出信件，否则您就得要透过 MUA 来帮你送信到邮件服务器上头去。 最常见的 MUA 像是 Mozilla 推出的 Thunderbird (雷鸟) 自由软件， 或者是 Linux 桌面 KDE 常见的 Kmail ，及 Windows 内件的 Outlook Express (OE) 等。 MUA 主要的功能就是收受邮件主机的电子邮件，以及提供用户浏览与编写邮件的功能！

2. MTA (Mail Transfer Agent)：

   MUA 帮用户传送邮件到邮件主机上，那这部邮件主机如果能够帮用户将这封信寄出去，那他就是一部邮件传送主机 (MTA) 啦！这个 MTA 就是『邮件传送代理人』的意思。也来顾名思义一下，既然是『传送代理人』， 那么使用者寄出的信，帮用户将属于该用户的信件收下时，就是找它 (MTA) 就对啦！基本上，MTA 的功能有这些：

   i. 收受信件：使用简单邮件传送协议(SMTP) MTA 主机最主要的功能就是：将来自客户端或者是其他 MTA 的来信收下来，这个时候 MTA 使用的是 Simple Mail Transfer Protocol (SMTP)，他使用的是 port 25 啦！

   ii. 转递信件： 如果该封信件的目的地并不是本身的用户，且该封信的相关数据符合使用 MTA 的权力， 那么咱们的 MTA 就会将该封信再传送到下一部主机上。这即是所谓的转递 (Relay) 的功能。

总之，我们一般提到的 Mail Server 就是 MTA 啦！而严格来说， MTA 其实仅是指 SMTP 这个协议而已。而达成 MTA 的 SMTP 功能的主要软件包括老牌的 sendmail，后起之秀的 postfix，还有 qmail 等等。底下我们来看看，那么在 MTA 上头还有哪些重要的功能。

3. MDA (Mail Delivery Agent)：

字面上的意思是『邮件递送代理人』的意思。事实上，这个 MDA 是挂在 MTA 底下的一个小程序， 最主要的功能就是：分析由 MTA 所收到的信件表头或内容等数据， 来决定这封邮件的去向。所以说，上面提到的 MTA 的信件转递功能，其实是由 MDA 达成的。举例来说，如果 MTA 所收到的这封信目标是自己，那么 MDA 会将这封信给他转到使用者的信箱 (Mailbox) 去， 如果不是呢？那就准备要转递出去了。此外，MDA 还有分析与过滤邮件的功能喔！举例来说：

i. 过滤垃圾信件： 可以根据该封邮件的表头资料，或者是特定的信件内容来加以分析过滤。例如某个广告信的主题都是固定的， 如『AV情色...』等等，那就可以透过 MDA 来过滤并去除该邮件。

ii. 自动回复： 如果您出差了导致某一段时间内无法立即回信时，就可以透过 MDA 的功能让邮件主机可以自动发出回复信件， 如此您的朋友就不会认为你太大牌！^_^ 各主要的 MTA 程序 (sendmail,postfix...) 都有自己的 MDA 功能，不过有些外挂的程序功能更强大，举例来说 procmail 就是一个过滤的好帮手，另外 Mailscanner + Spamassassion 也是可以使用的一些 MDA 喔。

4. Mailbox：

就是电子邮件信箱嘛！简单的说，就是某个账号专用的信件收受档案啰。我们的 Linux 系统默认的信箱都是放在 /var/spool/mail/使用者账号 中！ 若 MTA 所收到的信件是本机的使用者，MDA 就会将信件送到该 mailbox 当中去啰！

好了，那么来想一想，你如何透过 MUA 来将信件送到对方的邮件信箱 (Mailbox) 去呢？

- Step 0：取得某部 MTA 的权限：

  就如图 22.1-1 所示，我们本地端的 MUA 想要使用 MTA 来传出信件时， 当然需要取得 MTA 的权限。通常就是说：我们必须要向 MTA 注册一组可使用 email 的账号与密码才行。

- Step 1：使用者在 MUA 上编写信件后，传送至 MTA 上头：

  使用者在 MUA 上面编写信件，信件的数据主要有：

  - 信件标头：包括发件人与收件者的 email 地址，还有该封信件的主旨 (subject) 等；
  - 信件内容：就是你要跟对方说明的内容啦！ 编写完毕之后只要按下传送钮，该封信

就会送至你的 MTA 服务器上面了，注意：是你的 MTA 而不是对方的 MTA ！ 如果你确定可以使用该部 MTA，那么你的这封信就会被放置到 MTA 的队列 (queue) 当中并等待传送出去了。

- Step 2.1：如果该封信的目标是本地端 MTA 自己的账号

  你是可以寄信给你自己的，所以如果你的 MTA 收到该封信件的目标是自己的用户时，那就会透过 MDA 将这封信送到 Mailbox 去啰！

- Step 2.2：如果该封信目的为其他 MTA ，则开始转递 (Relay) 的流程：

  那如果这封信的目标是其他的主机呢？这个时候我们的 MTA 就会开始分析该封信是否具有合法的权限， 若具有权限时，则我们的 MDA 会开始进行邮件转递，亦即该封信件会透过我们的 MTA 向下一部 MTA 的 smtp (port 25) 发送出去。如果该封信件顺利的发送出去了，那么该封信件就会由队列当中移除掉了。

- Step 3：对方 MTA 服务器收受信件

  如果一切都没有问题的话，远程的 MTA 会收到我们 MTA 所发出的那封信，并将该信件放置到正确的使用者信箱当中， 等待使用者登入来读取或下载。

在这整个过程当中，你会发现你的信件是由我们的 MTA 帮忙发送出去的，此时 MTA 提供的协议是简单邮件传输协议 (Simple Mail Transfer Protocol, smtp)， 并且该封信最终是停留在对方主机的 MTA 上头！并不是你朋友的 MUA 上头啊！

Tips: 为何特别强调这一点？因为以前有个朋友有跟我说：『鸟哥啊，你要寄 email 给我的时候记得跟我讲， 那我下班前将计算机开着，以免你信寄不到我的信箱』，此时额头三条线突然跑出来～很不好意思～ 所以这里才要特别强调，你的 MUA 不必开着啦！要收信时再打开即可。



了解了传送信件时 MTA 需要启动 smtp (port 25) 之后，再来我们得要谈谈那这封信件对方要如何接收啊？

22.1.4 使用者收信时服务器端所提供的相关协议： MRA

那使用者如果想要收信时，当然也可以透过 MUA 直接来联机取得自己的邮件信箱内的数据啊！整个过程有点像底下这样：

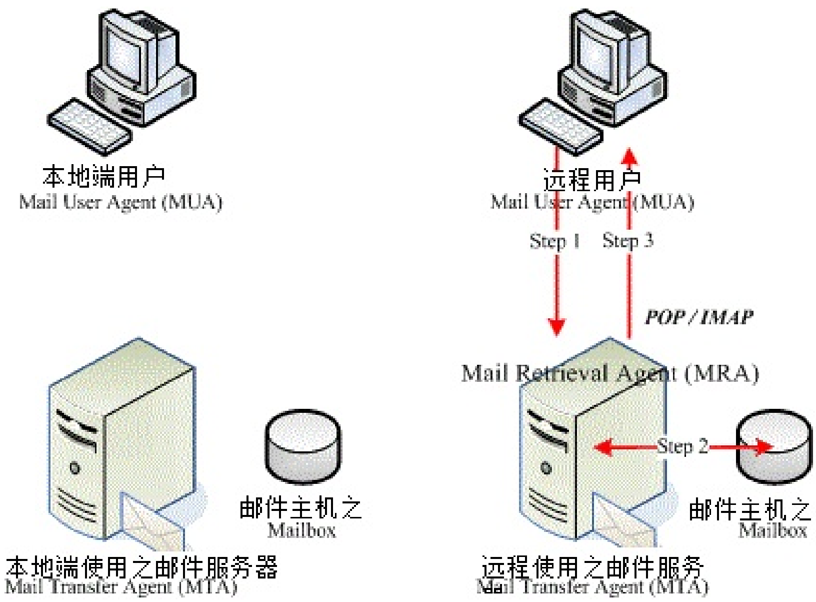


图 22.1-2、客户端透过 MRA 收回信件的流程示意图

在上述的图示中，多了一个邮件组件，那就是 MRA：

1. MRA (Mail Retrieval Agent)：

   使用者可以透过 MRA 服务器提供的邮政服务协议 (Post Office Protocol, POP) 来收下自己的信件， 也可以透过 IMAP (Internet Message Access Protocol) 协议将自己的信件保留在邮件主机上面， 并进一步建立邮件数据匣等进阶工作。也就是说，当客户端收受信件时，使用的是 MRA 的 POP3, IMAP 等通讯协议，并非 MTA 的 SMTP 喔！

我们先谈一谈 POP3 的收信方式吧：

1. MUA 透过 POP3 (Post Office Protocol version 3) 的协议连接到 MRA 的 port 110， 并且输入账号与密码来取得正确的认证与授权；
2. MRA 确认该用户账号/密码没有问题后，会前往该使用者的 Mailbox (/var/spool/mail/使用者账号) 取得使用者的信件并传送给用户的 MUA 软件上；
3. 当所有的信件传送完毕后，<u>用户的 mailbox 内的数据将会被删除</u>！

在上述的流程当中我们知道 MRA 必须要启动 POP3 这个协议才行，不过这个协议的收件方式比较有趣， 因为使用者收信是由第一封信件开始收下直到最后一封信件传输完毕为止。不过由于某些 MUA 程序撰写的问题，若有些邮件有病毒的可能性时，透过防病毒软件将可能导致该 MUA 软件的断线！ 如此一来由于传输没有完毕，因此 MRA 主机并不会将用户的信件删除。 此时如果使用者又再一次的按下接收按键，呵呵！原来已接收的信件又会重复收到，而没有收到的还是收不到！

这个时候或许你可以透过登入主机利用 mail 这个指令来处理你有问题的邮件， 或许换一种 MUA 也是个不错的思考方向，又或者暂时将防病毒软件关掉也是可以考虑的手段之一。 转头过来想一想，因为 POP3 的协议预设会将信件删除，那如果我今天在办公室将我的信收到办公室的计算机中， 当我回家时再度启动 MUA 时，是否能够收到已经被接收的信件？当然不行，对吧！

或许你需要更有帮助的协议，亦即 IMAP (Internet Messages Access Protocol) ， 这个协议可以让你将 mailbox 的数据转存到你主机上的家目录，亦即 /home/账号/ 那个目录下， 那你不但可以建立邮件数据匣，也可以针对信件分类管理，而且在任何一个可连上网络的地方你只要登入主机， 原本的信件就还是存在呐！真是好啊！

不过，使用 IMAP 时，用户的目录最好能够加点限制，例如利用 quota 来管理用户的硬盘权限， 否则因为信件都在主机上头，如果用户过多且误用时，你的硬盘空间会被吃光光喔！注意注意！

OK！透过上面的说明你要知道，要架设一部可以使用 MUA 进行收发信件的 MTA, MRA 服务器，你至少也需要启动 SMTP 以及 POP3 这两个协议才行！而这两个协议的启动程序并不相同， 所以架设上还是得要小心注意啊！

- pop3s, imap2 与 SMTP 的困扰

邮件数据在因特网上面传输时，透过的 SMTP, POP3, IMAP 等通讯协议，通通是明码传输的！尤其 POP3, IMAP 这两个通讯协议中，使用者必须要输入账号/密码才能收受信件！因为涉及帐密，所以当然加密这两个通讯协议的数据较佳！ 于是就有了 POP3s, IMAPs 通讯协议出现了！透过 SSL 加密嘛！那你会问，既然已经有 pop3s, imaps 了， 那有没有 smtps 呢？答案是，当然有！只不过没人用！

从图 22.1-1 及图 22.1-2 的流程来看，POP3, IMAP 只与 MRA 及自己的用户有关，因此你只要跟你的用户说，你服务器使用的 MRA 协议为何，通知你的用户改变即可，并不会影响到其他的服务器。 但是 MTA 就不同了！因为 MTA 必须与其他的 MTA 沟通，因此，若你使用了 smtps ，那么全世界与你的 MTA 沟通者， 通通需要改变为 smtps 通讯协议才行！这个工程实在太浩大了！目前还没有任何一家 ISP 有能力进行！ 所以，就造成目前没有 SMTPs 的协议啰。

那么难道你的数据就一定要是明码吗？那倒不见得～既然你的 MTA 无法加密，那么你就自己将邮件数据加密后，再交由 MTA 传送即可！这也是目前很多急需加密数据的邮件用户所使用的手段啦！^_^

22.1.5 Relay 与认证机制的重要性

当你需要 MTA 帮你将信寄送到下一部 MTA 去时，这个动作就称为邮件转递 (Relay) 啰，那就是图 22.1-1当中的 Step 2.2 那个动作啦。那么我们来想一想，如果『所有的人都可以藉由这一部 MTA 帮忙进行 Relay 时， 这个情况称之为 Open Relay 的动作』。当你的 MTA 发生 Open Relay 时，会有什么问题？ 问题可就大了！

当你的 MTA 由于设定不良的关系导致具有 Open Relay 的状况，加上你的 MTA 确实是连上因特网时， 由于因特网上面用 port scan 软件的闲人太多，你的 MTA 具有 Open Relay 的功能这件事情， 将会在短时间内就被很多人察觉，此时那些不法的广告信、色情垃圾信业者将会

利用你的这部 Open Relay MTA 发送他们的广告，所以你会发生的问题至少有：

- 你主机所在的网域正常使用的连接速度将会变慢，因为网络带宽都被广告、垃圾信吃光了；
- 你的主机可能由于大量发送信件导致主机资源被耗尽，容易产生不明原因当机之类的问题；
- 你的 MTA 将会被因特网社会定义为『黑名单』，从此很多正常的邮件就会无法收发；
- 你 MTA 所在的这个 IP 将会被上层 ISP 所封锁，直到你解决这个 Open Relay 的问题为止；
- 某些用户将会对你的能力产生质疑，对您公司或者是你个人将会有信心障碍！甚至可能流失客源；
- 如果你的 MTA 被利用来发黑函，你是找不到原发信者的，所以你这部 MTA 将会被追踪为最终站！

问题很大呦！所以啊，目前所有的 distributions 都一样，几乎都将 MTA 预设启动为仅监听内部循环接口 (lo) 而已，而且也将 Open Relay 的功能取消了。既然取消 Open Relay 的功能，那么怎么使用这部 MTA 的 Relay 来帮忙转信啊？呵呵！所以我们在上头才会一直说，你『必需』取得合法使用该 MTA 的权限啊！ 这也就是说，设定谁可以使用 Relay 的功能就是我们管理员的任务啦！通常设定 Relay 的方法有这几种：

- 规定某一个特定客户端的 IP 或网段，例如规定内部 LAN 的 192.168.1.0/24 可使用 Relay；
- 若客户端的 IP 不固定时 (例如拨接取得的非固定 IP) 可以利用认证机制来处理。
- 将 MUA 架设在 MTA 上面，例如 OpenWebMail 之类的 web 接口的 MUA 功能。

认证机制上面常见的有 SMTP 邮件认证机制，以及 SMTP after POP 两种，不论是哪一种机制， 基本上都是透过让使用者输入认证用的账号与密码，来确定他有合法使用该 MTA 的权限，然后针对通过认证者开启 Relay 的支持就是了。如此一来你的 MTA 不再启动 Open Relay ，并且客户端还是可以正常的利用认证机制来收发信件， 身为管理员的你可就轻松多啰！ ^_^

22.1.6 电子邮件的数据内容

看过上头的数据后，您应该对于 Mail server 有一些程度的认识了。再来要谈的是，那么一封 email 的内容有哪些部分呢？就跟人类社会的邮件有信封袋以及内部的信纸一样，email 也有所谓的标头 (header) 以及内容 (body) 两部份喔！

email 的标头部分 (类似邮件信封) 会有几个重要信息，包括：这封信来自那个 MTA、是由谁所发送出来的、要送给谁、 主旨为何等等，至于内容 (类似信封内的信纸) 则是发信者所填写的一些说明啰。如果你使用 dmtsai 的身份下达这个指令：

```
[dmtsai@www ~]$ echo "HaHa.." &#124; mail -s "from vbird" dmtsai
```

然后将自己的信箱内容叫出来，如下所示：

```
[dmtsai@www ~]$ cat /var/spool/mail/dmtsai
From dmtsai@www.centos.vbird  Mon Aug  8 18:53:32 2011  &lt;==发信者的 email
Return-Path: &lt;dmtsai@www.centos.vbird&gt;                  &lt;==这封信的来源
X-Original-To: dmtsai
Delivered-To: dmtsai@www.centos.vbird
Received: by www.centos.vbird (Postfix, from userid 2007)
        id 6D1C8366A; Mon,  8 Aug 2011 18:53:32 +0800 (CST) &lt;==邮件ID
# 这部份主要在讲这封 email 的来源与目标收件者 MTA 在哪里的信息～
Date: Mon, 08 Aug 2011 18:53:32 +0800     &lt;==收到信件的日期
To: dmtsai@www.centos.vbird               &lt;==收件者是谁啊！
Subject: from vbird                       &lt;==就是信件标题
User-Agent: Heirloom mailx 12.4 7/29/08
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
Message-Id: &lt;20110808105332.6D1C8366A@www.centos.vbird&gt; &lt;==给机器看的邮件ID
From: dmtsai@www.centos.vbird             &lt;==发信者是谁啊！

HaHa..
```

由原本的信件内容我们可以看到 email 确实是两部份，在标头部分记录了比较详细的收、发件者数据， 以及相关的来源、目标之 MTA 信息等等。但你要注意的是，那个『Received:...』那一行资料是『会变动的』， 如同前面谈到的 MX 标志，如果一封信由 MUA 传送到 MTA 在由 MTA 传送到 MX 主机后，才传送到最终的 MTA 时， 那么这个 Received: 的数据将会记录每一部经手过的 MTA 信息喔！所以你可以借着这个记录数据慢慢的找回这封信的传递方向呢！

此外，这个邮件的标头以及内容的分析部分，你还可以藉由某些分析软件来进行过滤， 这部份我们将在后头再慢慢的介绍给大家了解喔！ ^_^！您先知道一封邮件至少有这些数据，以后咱们再慢慢的解释啰！

22.2 MTA 服务器： Postfix 基础设定

可达成 MTA 的服务器软件非常多，例如我们的 CentOS 预设就提供了数十年老牌子的 sendmail (http://www.sendmail.org) 以及近期以来很热门的 Postfix (http://www.postfix.org)。虽然 sendmail 曾是最为广泛使用的 mail server 软件，但由于 sendmail 的配置文件太过于难懂，以及早期的程序漏洞问题导致的主机安全性缺失；加上 sendmail 将所有的功能都统合在 /usr/sbin/sendmail 这个程序当中，导致程序太大可能会有效能方面的疑虑等等， 所以新版的 CentOS 已经将预设的 mail server 调整为 postfix 啰！我们这里也主要介绍 postfix。当然啦，原理方面都一样，您也可以自己玩玩其他的 mail server。

22.2.1 Postfix 的开发

Postfix 是由 Wietse Zweitze Venema 先生(http://www.porcupine.org/wietse)所发展的。早期的 mail server 都是使用 sendmail 架设的，还真的是『仅此一家，绝无分号』！不过，Venema 博士觉得 sendmail 虽然很好用，但是毕竟不够安全，尤其效能上面并不十分的理想，最大的困扰是...sendmail 的配置文件 sendmail.cf 真的是太难懂了！对于网管人员来说，要设定好 sendmail.cf 这个档案，真不是人作的工作。

为了改善这些问题， Venema 博士就在 1998 年利用他老大在 IBM 公司的第一个休假年进行一个计划：『 设计一个可以取代 sendmail 的软件套件，可以提供网站管理员一个更快速、更安全、而且完全兼容于 sendmail 的 mail server 软件！』这个计划还真的成功了！ 而且也成功的使用在 IBM 内部，在 IBM 内可以说是完全取代了 sendmail 这个邮件服务器！在这个计划成功之后， Venema 博士也在 1998 年首次释出这个自行发展的邮件服务器，并定名为 VMailer。

不过，IBM 的律师却发现一件事，那就是 VMailer 这个名字与其他已注册的商标很类似， 这样可能会引起一些注册上面的困扰。为了避免这个问题，所以 Venema 博士就将这个邮件软件名称改为 Postfix ！『Post 有在什么什么之后』的意思，『fix 则是修订』的意思，所以 postfix 有 『在修订之后』的意思。

鸟哥个人认为， Venema 先生最早的构想并不是想要『创造一个全新的 Mail server 软件，而是想要制造一个可以完全兼容于 sendmail 的软件』，所以，Venema 先生认为他自行发展的软件应该是『改良 sendmail 的缺失』，所以才称为 Postfix 吧！取其意为： 『改良了 sendmail 之后的邮件服务器软件！』

所以啦， Postfix 设计的理念上面，主要是针对『想要完全兼容于 sendmail』所设计出来的一款『内在部分完全新颖』的一个邮件服务器软件。就是由于这个理念，因此 Postfix 改善了 sendmail 安全性上面的问题，改良了 mail server 的工作效率， 且让配置文件内容更具亲和力！因此，你可以轻易的由 sendmail 转换到 Postfix 上面！这也是当初 Venema 博士的最初构想啊！

就是基于这个构想，所以 Postfix 在外部配置文件案的支持度，与 sendmail 几乎没有两样，同样的支持 aliases 这个档案，同样的支持 ~/.forward 这个档案，也同样的支持 SASL 的 SMTP 邮件认证功能等等！ 所以，呵呵！赶紧来学一学怎样架设 Postfix 这个相当出色的邮件服务器吧！ ^_^

22.2.2 所需要的软件与软件结构

由于 CentOS 6.x 预设就是提供 postfix 的！所以根本无须调整啥咚咚～直接来使用吧！ 那么 postfix 有哪些重要的配置文件呢？他主要的配置文件都在 /etc/postfix/ 当中，详细的档案内容就让我们来谈谈：

- /etc/postfix/main.cf 这就是主要的 postfix 配置文件啰，几乎所有的设定参数都是在这个档案内规范的！ 这个档案预设就是一个完整的说明档了，你可以参考这个档案的内容就设定好属于你的 postfix MTA 呢！ 只要修改过这个档案，记得要重新启动 postfix 喔！
- /etc/postfix/master.cf 主要规定了 postfix 每个程序的运作参数，也是很重要的一个配置文件。不过这个档案预设已经很 OK 了，通常不需要更改他。
- /etc/postfix/access (利用 postmap 处理) 可以设定开放 Relay 或拒绝联机的来源或目标地址等信息的外部配置文件，不过这个档案要生效还需要在 /etc/postfix/main.cf 启动这个档案的用途才行。且设定完毕后需要以 postmap 来处理成为数据库档案呢！
- /etc/aliases (利用 postalias 或 newaliases 均可) 做为邮件别名的用途，也可以作为邮件群组的设定喔！

至于常见的执行档则有底下这些：

- /usr/sbin/postconf (查阅 postfix 的设定数据) 这个指令可以列出目前你的 postfix 的详细设定数据，包括系统默认值也会被列出来， 所以数据量相当的庞大！如果你在 main.cf 里面曾经修改过某些预设参数的话，想要仅列出非默认值的设定数据， 则可以使用『postconf -n』这个选项即可。
- /usr/sbin/postfix (主要的 daemon 指令) 此为 postfix 的主要执行档，你可以简单的使用他来启动或重新读取配置文件：

```
[root@www ~]# postfix check   &lt;==检查 postfix 相关的档案、权限等是否正确！
[root@www ~]# postfix start   &lt;==开始 postfix 的执行
[root@www ~]# postfix stop    &lt;==关闭 postfix
[root@www ~]# postfix flush   &lt;==强制将目前正在邮件队列的邮件寄出！
[root@www ~]# postfix reload  &lt;==重新读入配置文件，也就是 /etc/postfix/main.cf
```

  要注意的是，每次更动过 main.cf 后，务必重新启动 postfix，可简单的使用『postfix reload』即可。不过老实说，鸟哥还是习惯使用 /etc/init.d/postfix reload..

- /usr/sbin/postalias 设定别名数据库的指令，因为 MTA 读取数据库格式的档案效能较佳，所以我们都会将 ASCII 格式的档案重建为数据库。 在 postfix 当中，这个指令主要在转换 /etc/aliases 成为 /etc/aliases.db 啰！用法为：

```
[root@www ~]# postalias hash:/etc/aliases
# hash 为一种数据库的格式，然后那个 /etc/aliases.db 就会自动被更新啰！
```

- /usr/sbin/postcat 主要用在检查放在 queue (队列) 当中的信件内容。由于队列当中的信件内容是给 MTA 看的， 所以格式并不是一般我们人类看的懂的文字数据。所以这个时候你得要用 postcat 才可以看出该信件的内容。 在 /var/spool/postfix 内有相当多的目录，假设内有一个文件名为 /deferred/abcfile ， 那你可以利用底下的方式来查询该档案的内容喔：

```
[root@www ~]# postcat /var/spool/postfix/deferred/abcfile
```

- /usr/sbin/postmap 这个指令的用法与 postalias 类似，不过他主要在转换 access 这个档案的数据库啦！用法为：

```
[root@www ~]# postmap hash:/etc/postfix/access
```

- /usr/sbin/postqueue 类似 mailq 的输出结果，例如你可以输入『postqueue -p』看看就知道了！

整个 postfix 的软件结构大致上是这个样子的，接下来让我们先来简单的处理一下 postfix 的收发信件功能吧！

22.2.3 一个邮件服务器的设定案例

前面谈到 mail server 与 DNS 系统有很大的相关性，所以如果你想要架设一部可以连上 Internet 的邮件服务器时， 你必需要已经取得合法的 A 与 MX 主机名，而且最好反解也已经向您的 ISP 申请修改设定了， 这可是个大前提！不要忽略他！在底下的练习当中鸟哥以之前十九章 DNS 内的设定为依据，主要的参数是这样的：

- 邮件服务器的主要名称为： www.centos.vbird
- 邮件服务器尚有别名为 linux.centos.vbird 及 ftp.centos.vbird 也可以收发信件；
- 此邮件服务器已有 MX 设定，直接指向自己 (www.centos.vbird)
- 这个 www.centos.vbird 有个 A 的标志指向 192.168.100.254。

在实际的邮件服务器设定当中，上述的几个标志是很重要的，请自行参考 DNS 章节的介绍吧！底下就让我们来实际设定 postfix 服务器啰！

22.2.4 让 Postfix 可监听 Internet 来收发信件

在预设的情况下，CentOS 6.x 的 MTA 仅针对本机进行监听，不相信吗？测测看：

```
[root@www ~]# netstat -tlnp &#124; grep :25
Proto Recv-Q Send-Q Local Address   Foreign Address   State    PID/Program name
tcp        0      0 127.0.0.1:25    0.0.0.0:*         LISTEN   3167/master
```

所以如果你要对整个 Internet 开放的话，就得要努力的搞定几个简单的设定啰！而几乎所有的设定你都可已经由 /etc/postfix/main.cf 这个档案搞定！修改前你需要注意的项目有：

- 『 # 』符号是批注的意思；
- 所有设定值以类似『变量』的设定方法来处理，例如 myhostname = www.centos.vbird，请注意等号的两边要给予空格符喔， 且第一个字符不可以是空白，亦即『my..』要由行首写起；
- 可以使用『 $ 』来延伸使用变量设定，例如 myorigin = $myhostname，会等于 myorigin = www.centos.vbird；
- 如果该变量支持两个以上的数据，则使用空格符来分隔，不过建议使用逗号加空格符『, 』来处理。 例如： mydestination = $myhostname, $mydomain, linux.centos.vbird，意指 mydestination 支持三个数据内容之意。
- 可使用多行来表示同一个设定值，只要在第一行最后有逗号，且第二行开头为空格符，即可将数据延伸到第二行继续书写 (所以刚刚第二点才说，开头不能留白！)；
- 若重复设定某一项目，则以较晚出现的设定值为准！

要让你的 postfix 可以收发信件时，你必需要启动的设定数据有底下这些喔：

- myhostname：设定主机名，需使用 FQDN 喔

  这个项目在于设定你的主机名，且这个设定值会被后续很多其他的参数所引用，所以必须要设定正确才行。 你应该要设定成为完整的主机名。在鸟哥的这个练习当中，应该设定为： myhostname = www.centos.vbird 才对。 除了这个设定值之外，还有一个 mydomain 的设定项目，这个项目默认会取 $myhostname 第一个『.』之后的名称。 举例来说上头设定完毕后，预设的 mydomain 就是 centos.vbird 啰！你也可以自行设定他。

- myorigin ：发信时所显示的『发信源主机』项目

  这个项目在设定『邮件头上面的 mail from 的那个地址』， 也就是代表本 MTA 传出去的信件将以此设定值为准喔！如果你在本机寄信时忘记加上 Mail from 字样的话， 那么就以此值为准了。默认这个项目以 $myhostname 为主的，例如： myorigin = $myhostname

- inet_interfaces ：设定 postfix 的监听接口 (极重要)

在预设的情况下你的 Postfix 只会监听本机接口的 lo (127.0.0.1) 而已，如果你想要监听整个 Internet 的话， 请开放成为对外的接口，或者是开放给全部的接口，常见的设定方法为： inet_interfaces = all 才对！ 由于如果有重复设定项目时，会以最晚出现的设定值为准，所以最好只保留一组 inet_interfaces 的设定喔！

- inet_protocols ：设定 postfix 的监听 IP 协议

  预设 CentOS 的 postfix 会去同时监听 IPv4, IPv6 两个版本的 IP，如果你的网络环境里面仅有 IPv4 时，那可以直接指定 inet_protocols = ipv4 就会避免看到 :::1 之类的 IP 出现呦！

- mydestination ：设定『能够收信的主机名』 (极重要)

  这个设定项目很重要喔！因为我们的主机有非常多的名字，那么对方填写的 mail to 到底要写哪个主机名字我们才能将该信件收下？ 就是在这里规范的！也就是说，你的许多主机名当中，仅有写入这个设定值的名称才能作为 email 的主机地址。 在我们这个练习当中这部主机有三个名字，所以写法为： mydestination = $myhostname, localhost, linux.centos.vbird, ftp.centos.vbird

  如果你想要将此设定值移动到外部档案，那可以使用类似底下的作法： mydestination = /etc/postfix/local-host-names ，然后在 local-host-names 里面将可收信的主机名写入即可。一般来说，不建议你额外建立 local-host-names 这个档案啦， 直接写入 main.cf 即可说！特别留意的是，如果你的 DNS 里头的设定有 MX 标志的话，那么请将 MX 指向的那个主机名一定要写在这个 mydestination 内， 否则很容易出现错误讯息喔！一般来说，使用者最常发生错误的地方就在这个设定里头呢！

- mynetworks_style ：设定『信任网域』的一项指标

  这个设定值在规定『与主机在同一个网域的可信任客户端』的意思！举例来说，鸟哥的主机 IP 是 192.168.100.254，如果我相信整个局域网络内 (192.168.100.0/24) 的用户的话，那我可规定此设定值为『 subnet 』呐！ 不过，一般来说，因为底下的 mynetworks 会取代这个设定值， 所以不设定也没有关系喔！如果要设定的话，最好设定成为 host 即可 (亦即仅信任这部 MTA 主机而已)。

- mynetworks ：规定信任的客户端 (极重要)

  你的 MTA 能不能帮忙进行 Relay 与这个设定值最有关系！举例来说，我要开放本机与内部网域的 IP 时，就可以这样进行设定： mynetworks = 127.0.0.0/8, 192.168.100.0/24。 如果你想要以 /etc/postfix/access 这个档案来控制 relay 的用户时，那鸟哥可以建议你将上述的数据改写成这样： mynetworks = 127.0.0.0/8, 192.168.100.0/24, hash:/etc/postfix/access 然后你只要再建立 access 之后重整成数据库后，嘿嘿！就能够设定 Relay 的用户啰！

- relay_domains ：规范可以帮忙 relay 的下一部 MTA 主机地址

相对于 mynetworks 是针对『信任的客户端』而设定的，这个 relay_domains 则可以视为『针对下游 MTA 服务器』而设定的。举例来说，如果你这部主机是 www.niki.centos.vbird 的 MX 主机时， 那你就得要在 relay_domains 设定针对整个 niki.centos.vbird 这个领域的目标信件进行转递才行。 在预设的情况下，这个设定值是 $mydestination 而已啦。

你必需要注意的『Postfix 预设并不会转递 MX 主机的信件』，意思就是说：如果你有两部主机，一部是上游的 MTAup ，一部是下游的 MTAdown ，而 MTAdown 规范的 MX 主机是 MTAup，由 22.1.2 谈到的 DNS 的 MX 设定值与信件传递方向，我们知道任何想要寄给 MTAdown 主机的信件， 都会先经过 MTAup 来转递才行！此时如果那部 MTAup 没有开启帮 MTAdown 进行 relay 的权限时， 那么任何传给 MTAdown 的信件将『全部都被 MTAup 所退回』！从此 MTAdown 就无法收到任何信件了。

上一段的说明请您特别再想一想，因为如果你在大公司服务而且你的公司上、下游均有 mail server 时， 并且也有设定 MX 的状况下，嘿嘿！这个 relay_domains 就很重要啦！上游的 MTA 主机必需要启动这个设定。 一般来说除非你是某部 MTA 主机的 MX 源头，否则这个设定项目可以忽略不设定他。 而如果你想要帮你的客户端转递信件到某部特定的 MTA 主机时，这个设定项目也是可以设定的啦。 默认请您保留默认值即可。

- alias_maps ：设定邮件别名

  就是设定邮件别名的设定项目，只要指定到正确的档案去即可，这个设定值可以保留默认值啊：

在了解上述的设定后，以鸟哥的范例来看的话，鸟哥有更动过或注明重要的设定值以及相关档案是这样处理的：

```
[root@www ~]# vim /etc/postfix/main.cf
myhostname = www.centos.vbird          &lt;==约在第  77 行
myorigin = $myhostname                 &lt;==约在第  99 行
inet_interfaces = all                  &lt;==约在第 114 行，117 行要批注掉
inet_protocols = ipv4                  &lt;==约在第 120 行
mydestination = $myhostname, localhost.$mydomain, localhost,
  linux.centos.vbird, ftp.centos.vbird &lt;==约在第 165,166 行
mynetworks = 127.0.0.0/8, 192.168.100.0/24, hash:/etc/postfix/access &lt;==约在269行
relay_domains = $mydestination         &lt;==约在第 299 行
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases     &lt;==约在第 389, 400 行
# 其他的设定值就先保留默认值即可啊！

[root@www ~]# postmap hash:/etc/postfix/access
[root@www ~]# postalias hash:/etc/aliases
```

因为 main.cf 当中我们有额外加入两个外部配置文件 (mynetworks 及 alias_maps) ，所以才会额外进行 postmap 及 postalias。然后准备来启动啦！你可以这样处理喔：

```
# 1\. 先检查配置文件的语法是否有错误
[root@www ~]# /etc/init.d/postfix check   &lt;==没有讯息，表示没有问题。

# 2\. 启动与观察 port number
[root@www ~]# /etc/init.d/postfix restart
[root@www ~]# netstat -tlunp &#124; grep ':25'
Proto Recv-Q Send-Q Local Address  Foreign Address   State   PID/Program name
tcp        0      0 0.0.0.0:25     0.0.0.0:*         LISTEN  13697/master
```

很简单吧！这样就设定妥当了。假设你的防火墙已经处理完毕，那你的 Postfix 已经可以开放客户端进行转递，并且也可以收受信件啰！不过，到底在预设的情况下我们的 postfix 可以收下哪些信件？又可以针对哪些设定值的内容进行转递呢？这就得要参考下一小节的说明了。

22.2.5 信件传送流程与收信、relay 等重要观念

我想，您对于 MTA 的设定与收发信件应该有一定程度的概念了，不过要妥善设定好你的 MTA 时， 尤其是想要了解到整部 MTA 是如何收、发信件时，你最好还是要知道『我这部 MTA 如何接受来源主机所传来的信件，以及将信件转递到下一部主机去』的整个流程啊。 一般来说一封邮件传送会经过许多的流程为：

1. 送信端与收信端两部主机间会先经过一个握手 (ehlo) 的阶段，此时送信端被记录为发信来源(而不是 mail from)。 通过握手后就可以进行信件标头 (header) 的传送；
2. 此时收信端主机会分析标头的信息，若信件之 Mail to: 主机名为收信端主机，且该名称符合 <u>mydestination</u> 的设定，则该信件会开始被收下至队列，并进一步送到 mailbox 当中； 若不符合 <u>mydestination</u> 的设定，则终止联机且不会进行信件内容 (body) 的传送；
3. 若 Mail to: 主机名非为收信端本身，则开始进行转递 (relay) 的分析。
4. 转递过程首先分析该信件的来源是否符合信任的客户端 (这个客户端为步骤 1 所记录的发信主机喔)，亦即来源是否符合 <u>mynetworks</u> 的设定值，若符合则开始收下信件至队列中，并等待 MDA 将信件再转递出去，若不符 mynetworks 则继续下一步；
5. 分析信件来源或目标是否符合 <u>relay_domains</u> 的设定，若符合则信件将被收下至队列，并等待 MDA 将信件再转递出去；
6. 若这封信的标头数据都不合乎上述的规范，则终止联机，并不会接受信件的内容数据的。

整个流程有点像底下这样：

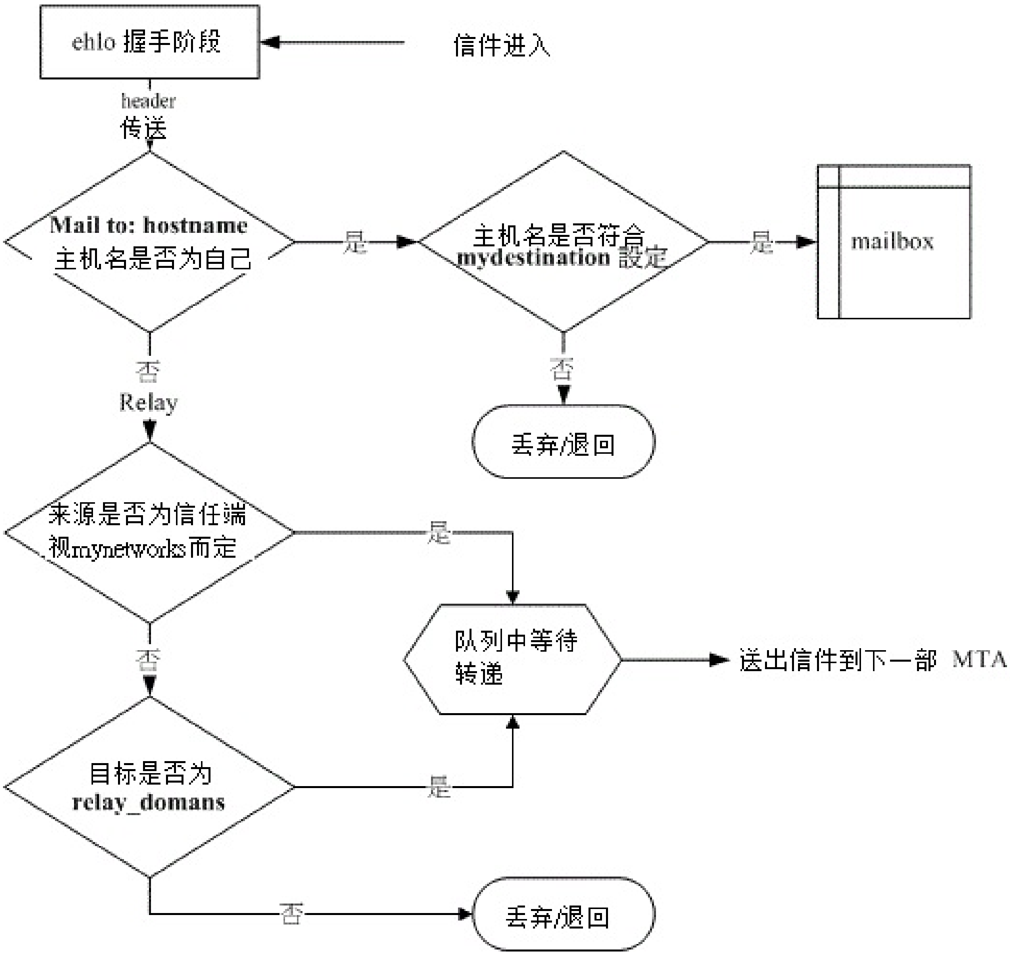


图 22.2-1、在本机 MTA 当中的信件分析过程

也就是说标头分析通过后，你的信件内容才会开始上传到主机的队列，然后透过 MDA 来处理该信件的流向。 而不是将信件完整的传送到主机后才开始分析的喔！这个得要特别注意呐！而透过上述的流程后， 在暂不考虑 access 以及 MDA 的分析机制中，一部 MTA 想要正确的收、发信件时，电子邮件必需要符合：

- 收信方面：必需符合底下需求：

  1. 发信端必需符合 $inet_interfaces 的设定；
  2. 信件标头之收件者主机名必需符合 $mydestination 的设定， 或者收件主机名需要符合 $virtual_maps (与虚拟主机有关) 的设定；
- 转递方面 (Relay)：必需符合底下需求：

  1. 发信端必需符合 $inet_interfaces 的设定；
  2. 发信端来源必需为 $mynetworks 的设定；发信端来源或信件标头之收件者主机名符合 $relay_domains 之设定内容。

同样的原理与想法你可以将他用在 sendmail 的设定当中喔！ ^_^！不过很多垃圾信却是藉由这个预设的收发管道来发送， 怎么说呢？请看底下的分析：

例题：在我的主机上面竟然发现这样的广告信，那就是『利用我的主机发送广告信给我自己！』为什么这样也可以呢？答：首先，你必需要熟悉一下上述的流程，在第 2 个步骤当中我们知道，当主机收到一封信且这封信的目标是自己， 并且也符合 mydestination 的设定时，该信件就会被收下来而不必验证客户端是否来自于 mynetworks 了。 所以说，任何人都可以用这个流程来寄信给你啊。不过，你的 MTA 并不是 open relay 啦，不会帮人家发送广告信的，不用担心。

例题：我的主机明明没有 Open relay ，但很多其他的 MTA 管理员发信给我，说我的主机的某个账号持续发送广告信， 但是我的主机明明没有那个账号啊！这是怎么回事？答：仔细看一下流程的步骤 1 与 2 ，确认该封信能否被收下来与发信端及收信端主机名有关。 而我们知道在邮件的 header 里面还有一个 mail from 的标头设定项目，这个标头设定是我们在查阅邮件时看到的『回邮地址』， 这个数据是可以伪造的！而且他与收发信件的数据无关！所以，您应该要告知对方 MTA 管理员， 请他提供详细的 log 数据，才能够判断该封信是否由你的主机所发送出去的。

一般来说，目前的广告业者很多都是利用这种欺敌的方式来处理的，所以您必需要请对方提供详细的 log file 数据以供查验才行喔！

22.2.6 设定邮件主机权限与过滤机制 /etc/postfix/access

基本上，指定了 Postfix 的 mynetworks 的信任来源就能够让使用者 relay 了，不过如果你依照鸟哥上述的方式 (22.2.4) 来设定你的 mynetworks 的话，那么我们还可以利用 access 这个档案来额外管理我们的信件过滤呢！基本的 access 语法为：

```
规范的范围或规则              Postfix 的动作 (范例如下)
IP/部分IP/主机名/Email等       OK/REJECT
```

假设你想要让 120.114.141.60 还有 .edu.tw 可以使用这部 MTA 来转递信件，且不许 av.com 以及 192.168.2.0/24 这个网域的使用时，可以这样做：

```
[root@www ~]# vim /etc/postfix/access
120.114.141.60        OK
.edu.tw             OK
av.com             REJECT
192.168.2\.          REJECT
# OK 表示可接受，而 REJECT 则表示拒绝。

[root@www ~]# postmap hash:/etc/postfix/access
[root@www ~]# ls -l /etc/postfix/access*
-rw-r--r--. 1 root root 19648 2011-08-09 14:05 /etc/postfix/access
-rw-r--r--. 1 root root 12288 2011-08-09 14:08 /etc/postfix/access.db
# 你会发现有个 access.db 的档案才会同步更新！这才是 postfix 实际读取的！
```

用这个档案设定最大的好处是，你不必重新启动 postfix，只要将数据库建立好， 立刻就生效了！这个档案还有其它的进阶功能，你可以自行进入该档案查阅就知道了。但是进阶设定还需要 main.cf 内的其他参数有设定才行！如果只有之前 $mynetworks 的设定值时，你只能利用 access.db 的方式来开放 relay 的能力而已。不过，至少他可以让我们的设定简化啰！ ^_^

22.2.7 设定邮件别名： /etc/aliases, ~/.forward

想一想，你的主机里面不是有很多系统账号吗？例如 named, apache, mysql...， 那么以这些账号执行的程序若有讯息发生时，他会将该讯息以 email 的方式传给谁？应该就是传给 named, apache... 等账号自己吧。 不过，你会发现其实这些系统账号的信息都是丢给 root！这是因为其他的系统账号并没有密码可登入，自然也就无法接收任何邮件了，所以若有邮件就给系统管理员啰。不过，咱们的 MTA 怎么知道这些信件要传给 root ？这就得要 aliases 这个邮件别名配置文件来处理啦！

- 邮件别名配置文件： /etc/aliases

在你的 /etc/aliases 档案内，你会发现类似底下的字样：

```
[root@www ~]# vim /etc/aliases
mailer-daemon:  postmaster
postmaster:     root
bin:            root
daemon:         root
....(底下省略)....
```

左边是『别名』右边是『实际存在的使用者账号或者是 email address』！ 就是透过这个设定值，所以让我们可以将所有系统账号所属的信件通通丢给 root 啊！好，我们现在将他扩大化，假如你的 MTA 内有一个实际的账号名称为 dmtsai ，这个使用者还想要使用 dermintsai 这个名称来收他的信件， 那么你可以这样做：

```
[root@www ~]# vim /etc/aliases
dermintsai:     dmtsai
# 左边是你额外所设定的，右边则是实际接收这封信的账号！

[root@www ~]# postalias hash:/etc/aliases
[root@www ~]# ll /etc/aliases*
-rw-r--r--. 1 root root  1535 2011-08-09 14:10 /etc/aliases
-rw-r--r--. 1 root root 12288 2011-08-09 14:10 /etc/aliases.db
```

从此之后不论是 dmtsai@www.centos.vbird 还是 dermintsai@www.centos.vbird 都会将信件丢到 /var/spool/mail/dmtsai 这个信箱当中喔！很方便吧！

- /etc/aliases 实际应用一：让一般账号可接收 root 的信

假设你是系统管理员，而你常用的一般账号为 dmtsai，但是系统出错时的重要信件都是寄给 root 啊， 偏偏 root 的信件不能被直接读取....所以说，如果能够将『给 root 的信也转寄一份给 dmtsai 』的话， 那就太好了！可以达到吗？当然可以！你可以这样做：

```
[root@www ~]# vim /etc/aliases
root:        root,dmtsai  &lt;==鸟哥建议这种写法！
# 信件会传给 root 与 dmtsai 这两个账号！

root:        dmtsai       &lt;==如果 dmtsai 不再是管理员怎办？
# 从此 root 收不到信了，都由 dmtsai 来接受！

[root@www ~]# postalias hash:/etc/aliases
```

上面那两行你可以择一使用，看看 root 要不要保留他的信件都可以的！鸟哥建议使用第一种方式，因为这样一来， 你的 dmtsai 可以收到 root 的信，且 root 自己也可以『备份』一份在他的信箱内，比较安全啦！

- /etc/aliases 实际应用二：发送群组寄信功能

想象一个情况，如果你是学校的老师，你虽然只带一班导生，但是『每年都一班』时，如果有一天你要将信发给所有的学生， 那在写 email 的标头时，可能就会头昏昏的了 (因为联络人名单太多了)！这个时候你可以这样做： (假设主机上学生的账号为 std001, std002...)

```
[root@www ~]# vim /etc/aliases
student2011:    std001,std002,std003,std004...

[root@www ~]# postalias hash:/etc/aliases
```

如此一来只要寄信到这部主机的 student2011 这个不存在的账号时，该封信就会被分别存到各个账号里头去， 管理上面是否很方便啊！ ^_^！事实上，邮件别名除了填写自己主机上面的实体用户之外，其实你可以填写外部主机的 email 喔！ 例如你要将本机的 dermintsai 那个不存在的用户的信件除了传给 dmtsai 之外，还要外传到 dmtsai@mail.niki.centos.vbird 时，可以这样做：

```
[root@www ~]# vim /etc/aliases
dermintasi:    dmtsai,dmtsai@mail.niki.centos.vbird

[root@www ~]# postalias hash:/etc/aliases
```

很方便吧！更多的功能就期待您自行发掘啰！

Tips: 在这本书里面，dmtai 的家目录并非在正规的 /home 底下，而是放置于 /winhome 当中 (参考第十六章的练习)，所以实际操作 mail 指令会出错！这是因为 SELinux 的关系！请参考 /var/log/messages 底下的建议动作去处理即可！



- 个人化的邮件转递： ~/.forward

虽然 /etc/aliases 可以帮我们达到邮件别名设定的好处，不过 /etc/aliases 是只有 root 才能修改的档案权限， 那我们一般使用者如果也想要进行邮件转递时，该如何是好？没关系，可以透过自己家目录下的 .forward 这个档案喔！ 举例来说，我的 dmtsai 这个账号所接收到的信件除了自己要保留一份之外，还要传给本机上的 vbird 以及 dmtsai@mail.niki.centos.vbird 时，那你可以这样做设定：

```
[dmtsai@www ~]$ vim .forward
# 注意！我现在的身份现在是 dmtsai 这个一般身份，而且在他的家目录下！
dmtsai
vbird
dmtsai@mail.niki.centos.vbird

[dmtsai@www ~]$ chmod 644 .forward
```

记得这个档案内容是一行一个账号 (或 email) ，而且权限方面非常重要：

- 该档案所在用户家目录权限，其 group、other 不可以有写入权限。
- .forward 档案权限，其 group、other 不可以有写入权限。

如此一来这封信就会开始转递啰！有趣吧！ ^_^

22.2.8 察看信件队列信息： postqueue, mailq

说实话，设定到此为止咱们的 postfix 应该可以应付一般小型企业之 mail server 的用途了！ 不过，有的时候毕竟因为网络的问题或者是对方主机的问题，可能导致某些信件无法送出而被暂存在队列中， 那我们如何了解队列当中有哪些邮件呢？还有，在队列当中等待送出的信件是如何送出的呢？

- 如果该封信在五分钟之内无法寄出，则通常系统会发出一封『警告信』给原发信者， 告知该封邮件尚无法被寄送出去，不过，系统仍会持续的尝试寄出该封邮件；
- 如果在四小时候仍无法寄出，系统会再次的发出警告信给原发信者；
- 如果持续进行五天都无法将信件送出，那么该封邮件就会退回给原发信者了！

当然啦，某些 MTA 已经取消了警告信的寄发，不过原则上，如果信件无法实时寄出去的话 MTA 还是会努力尝试 5 天的，如果接下来的 5 天都无法送出时，才会将原信件退回给发信者。 一般来说，如果 MTA 设定正确且网络没有问题时，应该是不可能会有信件被放在队列当中而传不出去的， 所以如果发现有信件在队列时，当然得要仔细的瞧一瞧啰！检查队列内容的方法可以使用 mailq ， 也可以使用 postqueue -p 来检查的：

```
[root@www ~]# postqueue -p
Mail queue is empty
```

若您的邮件如此显示时，恭喜您，没有什么问题邮件在队列当中。不过如果你将 postfix 关闭， 并尝试发一封信给任何人，那就可能会出现如下的画面啦：

```
[root@www ~]# /etc/init.d/postfix stop
[root@www ~]# echo "test" &#124; mail -s "testing queue" root
[root@www ~]# postqueue -p
postqueue: warning: Mail system is down -- accessing queue directly
-Queue ID- --Size-- ----Arrival Time---- -Sender/Recipient-------
5CFBB21DB       284 Tue Aug  9 06:21:58  root
                                         root
-- 0 Kbytes in 1 Request.
# 第一行就说明了无法寄出的原因为 Mail system is down 啦！
# 然后才出现无法寄出的信件信息！包括来源与目标喔！
```

输出的信息主要为：

- Queue ID：表示此封邮件队列的代表号 (ID)，这个号码是给 MTA 看的，我们看不懂不要紧；
- Size ：这封信有多大容量 (bytes) 的意思；
- Arrival Time：这封信什么时候进入队列的，并且可能会说明无法立即传送出去的原因；
- Sender/Recipient：送信与收信者的电子邮件啰！

事实上这封信是放置在 /var/spool/postfix 里面，由于信件内容已经编码为给 MTA 看的数据排列， 所以你可以使用 postcat 来读出原信件的内容喔！例如这样做 (注意看档名与 Queue ID 的对应！)：

```
[root@www ~]# cd /var/spool/postfix/maildrop
[root@www maildrop]# postcat 5CFBB21DB  &lt;==这个档名就是 Queue ID
*** ENVELOPE RECORDS 5CFBB21DB ***     &lt;==说明队列的编号啊
message_arrival_time: Tue Aug  9 14:21:58 2011
named_attribute: rewrite_context=local &lt;==分析 named (DNS) 的特性来自本机
sender_fullname: root                  &lt;==发信者的大名与 email
sender: root
recipient: root                        &lt;==就是收件者啰！
*** MESSAGE CONTENTS 5CFBB21DB ***     &lt;==底下则是信件的实际内容啊！
Date: Tue, 09 Aug 2011 14:21:58 +0800
To: root
Subject: testing queue
User-Agent: Heirloom mailx 12.4 7/29/08
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

test
*** HEADER EXTRACTED 5CFBB21DB ***
*** MESSAGE FILE END 5CFBB21DB ***
```

如此一来你就知道目前我们的 MTA 主机有多少未送出的信件，还有未送出信件的内容你也可以追踪的到了！ 很不错，对吧！不过，如果你想要我们的 postfix 立刻尝试将这些在队列当中的信件寄出去，那又该如何是好？ 你有几个作法啦，可以重新启动 postfix ，也可以透过 postfix 的动作来处理，例如：

```
[root@www ~]# /etc/init.d/postfix restart
[root@www ~]# postfix flush
```

鸟哥个人比较建议使用 postfix flush 啰！自行参考看看先！ ^_^！接下来，让我们先来处理一下收信的 MRA 服务器， 搞定后再来处理客户端的用户接口吧！

22.2.9 防火墙设置

因为整个 MTA 主要是透过 SMTP (port 25) 进行信件传送的任务，因此，针对 postfix 来说，只要放行 port 25 即可呦！ 修改一下 iptables.rule 吧！

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.rule
# 找到底下这一行，并且将它批注拿掉！
iptables -A INPUT -p TCP -i $EXTIF --dport  25  --sport 1024:65534 -j ACCEPT

[root@www ~]# /usr/local/virus/iptables/iptables.rule
```

这样就放行整个 Internet 对您服务器的 port 25 的读取啰！简单！搞定！

22.3 MRA 服务器： dovecot 设定

除非你想要架设 webmail 在你的 MTA 上头，否则，你的 MTA 收下了信件，你总得连上 MTA 去收信吧？那么收信要用的是哪个通讯协议？ 就是 22.1.4 里面谈到的 pop3 以及 imap 啰！这就是所谓的 MRA 服务器！我们的 CentOS 6.x 使用的是 dovecot 这个软件来达成 MRA 的相关通讯协议的！但由于 pop3/imap 还有数据加密的版本，底下我们就依据是否加密 (SSL) 来设定 dovecot 吧！

22.3.1 基础的 POP3/IMAP 设定

启动单纯的 pop3/imap 是很简单的啦，你得要先确定已经安装了 dovecot 这个软件。而这个软件的配置文件只有一个，就是 /etc/dovecot/dovecot.conf 。我们仅要启动 pop3/imap 而已，所以可以这样设定即可：

```
[root@www ~]# yum install dovecot
[root@www ~]# vim /etc/dovecot/dovecot.conf
# 找到底下这一行，大约是在第 25 行左右的地方，复制新增一行内容如下：
#protocols = imap pop3 lmtp
protocols = imap pop3

[root@www ~]# vim /etc/dovecot/conf.d/10-ssl.conf
ssl = no   &lt;==将第 6 行改成这样！
```

改完之后你就可以启动 dovecot 啰！并且检查看看 port 110/143 (pop3/imap) 有没有启动啊？

```
[root@www ~]# /etc/init.d/dovecot start
[root@www ~]# chkconfig dovecot on
[root@www ~]# netstat -tlnp &#124; grep dovecot
Proto Recv-Q Send-Q Local Address   Foreign Address   State    PID/Program name
tcp        0      0 :::110          :::*              LISTEN   14343/dovecot
tcp        0      0 :::143          :::*              LISTEN   14343/dovecot
```

耶！搞定！这样就可以提供使用者来收信件啦！真是不错啊！不过记得喔，这里只提供基本的明码 pop3/imap 传输而已， 如果想要启动其他如 pop3s (传输加密机制) 协议时，就得要额外的设定啰！

22.3.2 加密的 POP3s/IMAPs 设定

如果担心数据在传输过程会被窃取，或者是你的登入信息 (账号与密码) 在使用 pop3/imap 时会被窃听， 那么这个 pop3s/imaps 就显的重要啦！与之前的 Apache 相似的，其实我们都是透过 openssl 这个软件提供的 SSL 加密机制来进行数据的加密传输。方式很简单呢！预设的

情况下，CentOS 已经提供了 SSL 凭证范例文件给我们使用了。 如果你一点都不想要使用预设的凭证，那么我们就来自己建一个吧！

```
# 1\. 建立凭证：到系统提供的 /etc/pki/tls/certs/ 目录下建立所需要的 pem 凭证档：
[root@www ~]# cd /etc/pki/tls/certs/
[root@www certs]# make vbirddovecot.pem
....(前面省略)....
Country Name (2 letter code) [XX]:TW
State or Province Name (full name) []:Taiwan
Locality Name (eg, city) [Default City]:Tainan
Organization Name (eg, company) [Default Company Ltd]:KSU
Organizational Unit Name (eg, section) []:DIC
Common Name (eg, your name or your server's hostname) []:www.centos.vbird
Email Address []:dmtsai@www.centos.vbird

# 2\. 因为担心 SELinux 的问题，所以建议将 pem 档案放置到系统默认的目录去较佳！
[root@www certs]# mv vbirddovecot.pem ../../dovecot/
[root@www certs]# restorecon -Rv ../../dovecot

# 3\. 开始处理 dovecot.conf，只要 pop3s, imaps 不要明码传输的咯！
[root@www certs]# vim /etc/dovecot/conf.d/10-auth.conf
disable_plaintext_auth = yes  &lt;==第 9 行改成这样！取消批注！

[root@www certs]# vim /etc/dovecot/conf.d/10-ssl.conf
ssl = required                                &lt;==第 6 行改成这样
ssl_cert = &lt;/etc/pki/dovecot/vbirddovecot.pem &lt;==12, 13 行变这样
ssl_key =  &lt;/etc/pki/dovecot/vbirddovecot.pem

[root@www certs]# vim /etc/dovecot/conf.d/10-master.conf
  inet_listener imap {
    port = 0     &lt;== 15 行改成这样
  }
  inet_listener pop3 {
    port = 0     &lt;== 36 行改成这样
  }

# 4\. 处理额外的 mail_location 设定值！很重要！否则网络收信会失败：
[root@www certs]# vim /etc/dovecot/conf.d/10-mail.conf
mail_location = mbox:~/mail:INBOX=/var/mail/%u &lt;==第 30 行改这样

# 5\. 重新启动 dovecot 并且观察 port 的变化：
[root@www certs]# /etc/init.d/dovecot restart
[root@www certs]# netstat -tlnp &#124; grep dovecot
Proto Recv-Q Send-Q Local Address  Foreign Address   State    PID/Program name
tcp        0      0 :::993         :::*              LISTEN   14527/dovecot
tcp        0      0 :::995         :::*              LISTEN   14527/dovecot
```

最终你看到的 993 是 imaps 而 995 则是 pop3s 啰！这样一来，你收信的时候，输入的账号密码就不怕被窃听了！ 反正是加密后的资料啰！很简单吧！

22.3.3 防火墙设置

因为上面的练习中，我们将 pop3/imap 关闭，转而打开 pop3s/imaps 了，因此防火墙启动的埠口会不一样！ 请依据您实际的案例来设定你所需要的防火墙才好。我们这里主要是开放 993, 995 两个埠口呦！ 处理的方法与 22.2.9 相当类似：

```
[root@www ~]# vim /usr/local/virus/iptables/iptables.rule
# 大约在 180 行左右，新增底下两行去！
iptables -A INPUT -p TCP -i $EXTIF --dport 993  --sport 1024:65534 -j ACCEPT
iptables -A INPUT -p TCP -i $EXTIF --dport 995  --sport 1024:65534 -j ACCEPT

[root@www ~]# /usr/local/virus/iptables/iptables.rule
```

如果你的 pop3/imap 还是决定不加密的话，请将上面的 993/995 改成 143/110 即可！

22.4 MUA 软件：客户端的收发信软件

设定 Mail server 不是拿来好看的，当然是要好好的应用他啰！应用 mail server 有两种主要的方式，你可以直接登入 Linux 主机来操作 MTA ，当然也可以透过客户端的 MUA 软件来收发信件，底下我们分别介绍这两种方式啰！

22.4.1 Linux mail

在 Unix like 的操作系统当中都会存有一支可以进行收发信件的软件，那就是『 mail 』这个指令。这个指令是由 mailx 这个软件所提供的，所以您得要先安装这个软件才行。另外，由于 mail 是 Linux 系统的功能，所以即使你的 port 25 (smtp) 没有启动，他还是可以使用的，只是该封邮件就只会被放到队列，而无法寄出去啰！^_^！底下我们来谈一谈最简单的 mail 用法吧

- 用 mail 直接编辑文字邮件与寄信

mail 的用法很简单，就是利用『 mail [email address] 』的方式来将信件寄出去， 那个 [email address] 可以是对外的邮件地址，也可以是本机的账号。如果是本机账号的话，可以直接加账号名称即可。 例如：『 mail root 』或『 mail somebody@his.host.name 』 。如果是对外寄信的时候， 信件预设的『 Mail from 』就会填写 main.cf 内那个 myorigin 变数的主机名啰！ 先来试看看吧！寄给 dmtsai@www.centos.vbird 先：

```
[root@www ~]# mail dmtsai@www.centos.vbird
Subject: Just test        &lt;==这里填写信件标题
This is a test email.     &lt;==底下为信件的内容！
bye bye !
.                         &lt;==注意，这一行只有小数点！代表结束输入之意！
```

这样就可以将信件寄出去了！另外，早期的 mail server 是可以接受 IP 寄信的，举例来说：mail dmtsai@[192.168.100.254] ，记得 IP 要用中括号包起来。不过由于受到垃圾邮件的影响， 现在这种方式几乎都无法成功的将信件寄出了。

- 利用已经处理完毕的『纯文本档』寄出信件

这可不是『附件夹带』的方式！因为在 mail 这个程序里面编辑信件是个很痛苦的差事， 你不能够按上下左右键来回到刚刚编辑有错误的地方，很伤脑筋。此时我们可以透过标准输入来处理！ 如果你忘记『 < 』代表的意义，请回到基础篇的第十一章 bash shell中的数据流重导向瞧瞧先！举例来说你要将家目录的 .bashrc 寄给别人，可以这样做：

```
[root@www ~]# mail -s 'My bashrc' dmtsai &lt; ~/.bashrc
```

- 开始查阅接收的信件

寄信还比较简单，那么收信呢？同样的收信还是使用 mail。直接在提示字符之后输入 mail 时，会主动的捉取使用者在 /var/spool/mail 底下的邮件信箱 (mailbox)，例如我 dmtsai 这个账号在输入 mail 后，就会将 /var/spool/mail/dmtsai 这个档案的内容读出来并显示到屏幕上，结果如下：

```
# 注意喔！底下的身份使用的是 dmtsai 这个用户来操作 mail 这个指令的呦！
[dmtsai@www ~]$ mail
Heirloom Mail version 12.4 7/29/08\.  Type ? for help.
"/var/spool/mail/dmtsai": 10 messages 10 new &lt;==信箱来源与新信件数
&gt;N  1 dmtsai@www.centos.vb  Mon Aug  8 18:53  18/579   "from vbird"
....(中间省略)....
 N  9 root                  Tue Aug  9 15:04  19/618   "Just test"
 N 10 root                  Tue Aug  9 15:04  29/745   "My bashrc"
&  &lt;==这个是 mail 软件的提示字符，可以输入 ? 来察看可用指令
```

在上面的画面中，显示 dmtsai 有一封信，且会附上该信件的发信者与标题及收信时间等。你可以用的指令有这些：

- 读信：(直接按 Enter 或输入数字后 enter) 有看到『 > 』那个符号吧！那表示目前 mail 所在的邮件位置，你可以直接输入 Enter 即可看到该封信件的内容！ 另外，你也可以在『&』之后的光标位置输入号码，就可以看该封信件的内容了！(注：如果持续按 Enter，则会自『 > 』符号所在的邮件逐次向后读取每封信件内容！)
- 显示标题：(直接数入 h 或输入 h 数字) 例如有 100 封信，要看 90 封左右的信件标题，就输入『 h90 』即可。
- 回复邮件：(直接输入 R) 如果要回复目前『 > 』符号所在的邮件，直接按下『 R 』即可进入刚刚前面介绍过的 mail 文字编辑画面啰！你可以编辑信件后传回去啰！
- 删除邮件：(输入 d 数字) 按下『 d## 』即可删除邮件！例如我要删除掉第 2 封邮件，可以输入『 d2 』如果是要删除第 10-50 封邮件，可以输入『 d10-50 』来删除喔！请记得，如果有删除邮件的话，离开 mail box 时，要使用『 q 』才行！
- 储存邮件到档案：(输入 s 数字 文件名) 如果要将邮件资料存下来，可以输入『 s ## filename 』，例如我要将上面第 10 封邮件存下来，可以输入『 s 10 text.txt 』即可将第一封邮件内容存成 text.txt 这个档案！
- 离开 mail：(输入 q 或 x) 要离开 mail 可以输入 q 或者是 x，请注意『输入 x 可以在不更动 mail box 的情况下离开 mail 程序，不管你刚刚有没有使用 d 删除数据；使用 q 才会将删除的数据移除。』也就是说，如果你不想更动 mail box 那就使用 x 或 exit 离开，如果想要使刚刚移除的动作生效，就要使用 q 啦！
- 请求协助： 关于 mail 更详细的用法可以输入 help 就可以显现目前的 mail 所有功能！

上面是简易的 mail 收信功能！不过，我们曾经将信件转存下来的话，那该如何读取该信件呢？例如读取刚刚记录的 text.txt 邮件信箱。其实可以简单的使用这个方式来读取：

```
[dmtsai@www ~]$ mail -f ~/text.txt
```

- 以『附件夹带』的方式寄信

前面提到的都是信件的内容，那么有没有可能以『附件』的方式来传递档案？是可以的，不过你需要 uuencode 这个指令的帮忙，在 CentOS 当中这个指令属于 sharutils ，请先利用 yum 来安装他吧！接下来你可以这样使用：

```
[root@www ~]# [利用 uuencode 编码 ] &#124; [利用 mail 寄出去]
[root@www ~]# uuencode [实际档案] [信件中的檔名] &#124; mail -s '标题' email

# 1\. 将 /etc/hosts 以附件夹带的方式寄给 dmtsai
[root@www ~]# uuencode /etc/hosts myhosts &#124; mail -s 'test encode' dmtsai
```

这样就能寄出去了，不过，如果收下这封信件呢？同样的我们得要透过译码器来解碼啊！ 你得先将该档案存下来，然后这样做：

```
# 底下的身份可是 dmtsai 这个用户喔！
[dmtsai@www ~]$ mail
Heirloom Mail version 12.4 7/29/08\.  Type ? for help.
"/var/spool/mail/dmtsai": 11 messages 1 new 8 unread
    1 dmtsai@www.centos.vb  Mon Aug  8 18:53  19/590   "from vbird"
....(中间省略)....
 U 10 root                  Tue Aug  9 15:04  30/755   "My bashrc"
&gt;N 11 root                  Tue Aug  9 15:12  29/1121  "test encode"
& s 11 test_encode
"test_encode" [New file] 31/1141
& exit

[dmtsai@www ~]$ uudecode test_encode -o decode
                             加密檔         输出档
[dmtsai@www ~]$ ll *code*
-rw-r--r--. 1 dmtsai dmtsai  380 Aug  9 15:15 decode      &lt;==译码后的正确数据
-rw-rw-r--. 1 dmtsai dmtsai 1121 Aug  9 15:13 test_encode &lt;==内文会有乱码
```

虽然 mail 这个指令不是挺好用的，不过至少他可以提供我们在 Linux 纯文本模式下的一个简单的收发信件功能！ 不过，目前有个更棒的替代方案，那就是 mutt 这玩意儿啰！

22.4.2 Linux mutt

mutt 除了可以仿真 mail 这个指令之外，他还能够透过 pop3/imap 之类的协议去读取外部的信件喔！所以这家伙真的很不赖！ 让我们来玩玩 mutt 这个好物吧！在开始底下的动作前，请使用 yum install mutt 安装好它吧！

- 直接以 mutt 进行寄送信件的动作：含快速附件夹带文件

mutt 的功能也很多，我们先来看看 mutt 的基本语法好了，再来开始进行练习吧！

```
[root@www ~]# mutt [-a 附加檔] [-i 内文档] [-b 秘密副本] [-c 一般副本] \
&gt;  [-s 信件标题] email地址
选项与参数：
-a 附加檔：后面就是你想要传送给朋友的档案，是附加档案，不是信件内容喔！
-i 内文档：就是信件的内文部分，先编写成为档案而已；
-b 秘密副本：原收件者不知道这封信还会寄给后面的那个秘密副本收件者；
-c 一般副本：原收件者会看到这封信还有传给哪位收件者；
-s 信件标题：这还需要解释吗？这封信的标头！
email地址：就是原收件者的 email 啰！

# 1\. 直接在线编写信件，然后寄给 dmtsai@www.centos.vbird 这个用户
[root@www ~]# mutt -s '一封测试信' dmtsai@www.centos.vbird
/root/Mail 不存在。建立吗？ ([yes]/no): y  &lt;==第一次用才会出现这个讯息
To: dmtsai@www.centos.vbird
Subject: 一封测试信
随便写写！随便看看～！  &lt;==会进入 vi 画面编辑！很棒！

y:寄出  q:中断  t:To  c:CC  s:Subj  a:附加档案  d:叙述  ?:求助 &lt;==按下 y 寄出
    From: root &lt;root@www.centos.vbird&gt;
      To: dmtsai@www.centos.vbird
      Cc:
     Bcc:
 Subject: 一封测试信
Reply-To:
     Fcc: ~/sent
Security: 清除

-- 附件
- I     1 /tmp/mutt-www-2784-0      [text/plain, 8bit, utf-8, 0.1K]

# 2\. 将 /etc/hosts 当成信件内容寄给 dmtsai@www.centos.vbird 这个用户
[root@www ~]# mutt -s 'hosts' -i /etc/hosts dmtsai@www.centos.vbird
# 记得最终在 vim 底下要按下 :wq 来储存寄出喔！
```

与 mail 在线编写文字不一样，mutt 竟然会呼叫 vi 让你去编辑你的信件！如此一来，当然不需要预先编写信件内文了！ 这真是让人感到非常的开心啊！而且整个画面非常的直觉化！相当容易处理呢！那么如果需要附件夹带呢？尤其是夹带 binary program 时，可以这样做：

```
# 1\. 将 /usr/bin/passwd 当成附件夹带，寄给 dmtsai@www.centos.vbird 用户
[root@www ~]# mutt -s '附件' -a /usr/bin/passwd -- dmtsai@www.centos.vbird
To: dmtsai@www.centos.vbird
Subject: 附件
不过是个附件测试！

y:寄出  q:中断  t:To  c:CC  s:Subj  a:附加档案  d:叙述  ?:求助  &lt;==按 y 送出
    From: root &lt;root@www.centos.vbird&gt;
      To: dmtsai@www.centos.vbird
      Cc:
     Bcc:
 Subject: 附件
Reply-To:
     Fcc: ~/sent
Security: 清除

-- 附件
- I   1 /tmp/mutt-www-2839-0       [text/plain, 8bit, utf-8, 0.1K] &lt;==内文档
 A   2 /usr/bin/passwd          [applica/octet-stre, base64, 31K] &lt;==附加檔
```

看到上表中的附件底下那两行吗？I 代表的是直接附在信件内的内文，A 才是附加档案！这样看懂了吗？ 不过你想要使用 mutt 来附加档案时，必须要有底下的注意事项才行：

- 『 -a filename 』这个选项必须是在指令的最后面，如果上述的指令改写成：『 mutt -a

/usr/bin/passwd -s "附件" ... 』就不行！会失败的！

- 在文件名与 email 地址之间需要加上两个连续减号『 -- 』才行！如同上面测试的指令模样！

- 以 mutt 来读不同通讯协议的信箱

与 mail 比较之下，mutt 可以直接透过网络的 pop3, imap 等通讯协议来读信，是相当优秀的一个功能吻！ 至少鸟哥觉得真好用！底下同样的，先来瞧瞧可以使用的语法，然后再来看看一些练习。

```
[root@www ~]# mutt [-f 信箱位置]
选项与参数：
-f 信箱位置：如果是 imaps 的信箱，可以这样：『 -f imaps://服务器的IP 』

# 1\. 直接用 dmtsai 的身份读取本机的信箱内容：
[dmtsai@www ~]$ mutt
q:离开  d:删除  u:反删除  s:储存  m:信件  r:回复  g:群组  ?:求助
....(中间省略)....
  11 O + Aug 09 root            (  12) test encode
  12 O + Aug 09 root            (   1) 一封测试信
  13 O + Aug 09 root            (   8) hosts
 14 O + Aug 09 root            ( 604) 附件

---Mutt: /var/spool/mail/dmtsai [Msgs:14 Old:11 74K]---(date/date)-------(all)--

# 2\. 在上面的信件 14 号内容反白后，直接按下 Enter 会出现如下画面！：
i:离开  -:上一页  &lt;Space&gt;:下一页 v:显示附件。 d:删除  r:回复  j:下一个 ?:求助
Date: Tue, 9 Aug 2011 15:24:34 +0800
From: root &lt;root@www.centos.vbird&gt;
To: dmtsai@www.centos.vbird
Subject: 附件
User-Agent: Mutt/1.5.20 (2009-12-10)

[-- 附件 #1 --]
[-- 种类：textplain，编码：8bit，大小：0.1K --]

不过是个附件测试！          &lt;==信件的内文部分

[-- 附件 #2: passwd --]     &lt;==说明信件的附件夹带部分
[-- 种类：applicationoctet-stream，编码：base64，大小：41K --]

[-- application/octet-stream 尚未支持 （按 'v' 来显示这部份） --]

-O +- 14/14: root                附件                              -- (all)

# 3\. 在上面画面按下 v 后，会出现相关的附件数据：
q:离开  s:储存  &#124;:管线  p:显示  ?:求助
  I     1 &lt;no description&gt;                     [text/plain, 8bit, utf-8, 0.1K]
 A     2 passwd                           [applica/octet-stre, base64, 41K]
# 反白处按下 s 就能够储存附加档案啰！
```

最后离开时，一直按下 q ，然后参考出现的信息来处理即可这就是本机信件的收信方式！非常简单！ 附加档案的储存方面也很容易，真是非常开心啊！那如果是外部信箱呢？举例来说，我用 root 的身份去收 dmtsai 的 imaps 信件，会是怎样的情况呢？

```
# 1\. 在服务器端必须要让 mail 这个群组能够使用 dmtsai 的家目录，所以要这样：
[dmtsai@www ~]$ chmod a+x ~

# 2\. 开始在客户端登入 imaps 服务器取得 dmtsai 的新邮件与邮件文件夹
[root@www ~]# mutt -f imaps://www.centos.vbird
q:离开  ?:求助
这个验证属于：
   www.centos.vbird  dmtsai@www.centos.vbird
   KSU
   DIC
   Tainan  Taiwan  TW

这个验证的派发者：
   www.centos.vbird  dmtsai@www.centos.vbird
   KSU
   DIC
   Tainan  Taiwan  TW

这个验证有效
   由 Tue, 9 Aug 2011 06:45:32 UTC
     至 Wed, 8 Aug 2012 06:45:32 UTC
SHA1 Fingerprint: E86B 5364 2371 CD28 735C 9018 533F 4BC0 9166 FD03
MD5 Fingerprint: 54F5 CA4E 86E1 63CD 25A9 707E B76F 5B52

-- Mutt: SSL Certificate check (certificate 1 of 1 in chain)
(1)不接受，(2)只是这次接受，(3)永远接受 &lt;==这里要填写 2 或 3 才行！
在 www.centos.vbird 的使用者名称：dmtsai
dmtsai@www.centos.vbird 的密码：
```

最终在密码设定正确后，你就会看到刚刚我们所看到的信件了！不过要注意的是，如果你的用户家目录在非正规目录， 那么可能会出现 SELinux 的错误，这时就得要重新修订一下你的 SELinux 安全本文的类型啰！ 如此一来，我们就直接以文本模式来取得网络邮件信箱！这实在是非常方便的一件事！只是没有图文并茂而已！ ^_^

22.4.3 Thunderbird 好用的跨平台 (Windows/Linux X) 软件

自由软件最大的好处之一就是该软件大多可以进行移植，也就是在任何操作系统上面几乎都能够执行该软件的意思。 因此学习自由软件的好处就是，你不必因为转换操作系统而学习不同的操作环境！MUA 也有自由软件！那就是 Mozilla 基金会推出的 ThunderBird (雷鸟) 这个好用的咚咚，你可以在底下的网址上面找到繁体中文的软件：

- http://moztw.org/thunderbird/

有鉴于目前客户端还是以 Windows 操作系统为大宗，所以底下的说明主要是在 Windows 7 上头的安装与设定为主。目前 (2011/08) 最新的 Thunderbird 已经出到 5.x 了，所以鸟哥以繁体中文的 5.x 为范例来介绍啰。下载完毕的安装过程鸟哥省略了，因为一直下一步而已。鸟哥直接跳到第一次启动 Thunderbird 的介绍，希望对大家有帮助呦！鸟哥是以 dmtsai@www.centos.vbird 这个账号为范例来说明的呦！初次启动会出现下图：



图 22.4-1、第一次启动 Thunderbird 的示意图

由于是第一次启动，所以 thunderbird 里面没有任何识别数据。此时你可以填写你要在 email 上面让人家看到的资料， 以及包括你登入远程信箱的账号密码等信息。上图鸟哥的昵称为『鸟哥哥』，而 Email 是要给收件者看到的，密码当然就是自己的不外流～填完之后按下『继续』吧！

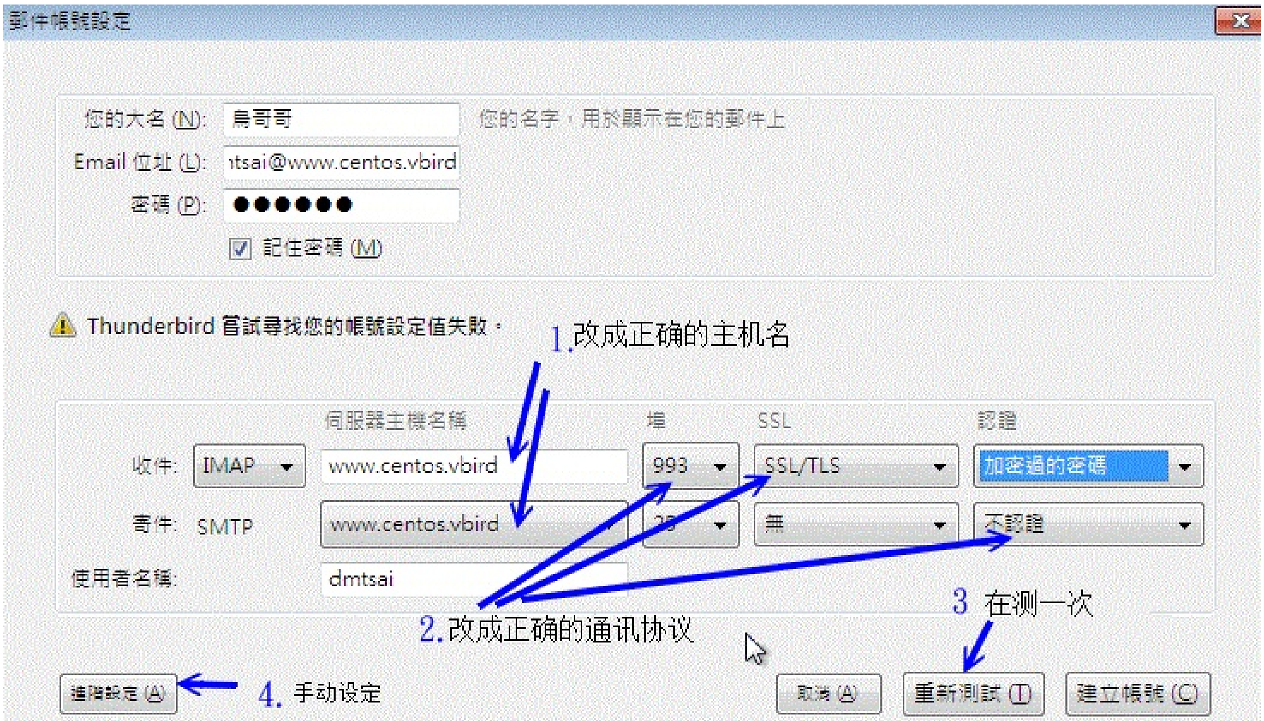


图 22.4-2、Thunderbird 主动的以用户信息尝试登入服务器

由于刚刚图 22.4-1 有输入账号与密码信息，因此，在这一个步骤中，Thunderbird 会主动的尝试登入远程信箱！ 不过，好像会抓取错误的信息的样子。如果真的抓错了，请修改箭头 1 指的服务器主机名，以及通讯协议的相关设定值， 按下『重新侦测』，确定捉到的数据是正确了，再按下『建立账号』或『进阶设定』(箭头4指的地方)即可！ 如果你很好奇进阶设定里面有啥，点选箭头 4 指的地方，会出现如下的详细资料：

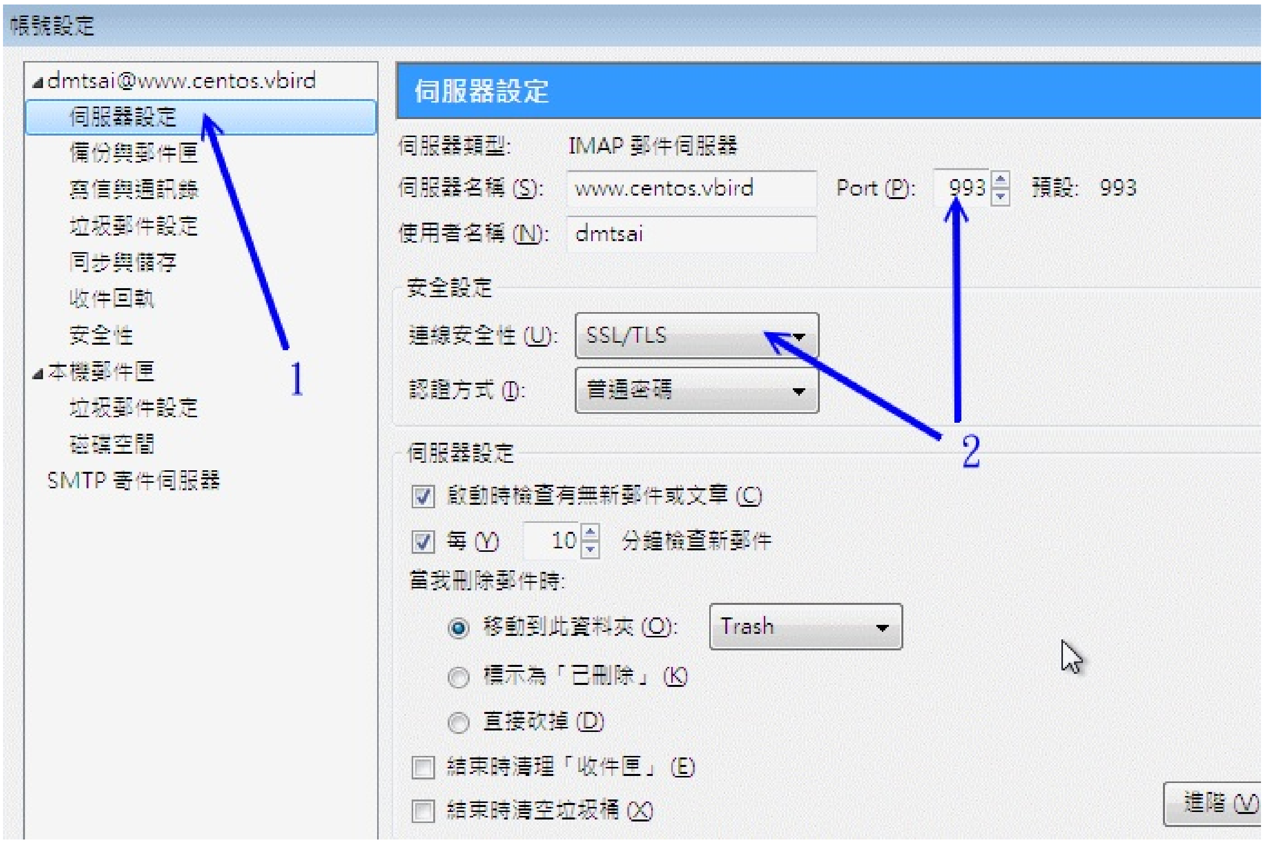


图 22.4-3、手动修改账号的相关参数

如上图所示，点选服务器设定项目，然后去查阅一下收信的服务器设定是否正确？若正确的话，就按下确定吧！ 然后会出现如下的图示，要你确定是否使用 Thunderbird 作为默认的电子邮件收发软件就是了！直接点确定进入下个步骤吧！



图 22.4-4、建立默认的 MUA 软件示意图

由于 Thunderbird 会尝试使用你输入的账号密码去登入远程服务器的 imaps 服务，所以就会出现如下图一般的凭证取得示意， 这时要按啥？当然是确认永久储存该凭证嘛！很简单的啊！

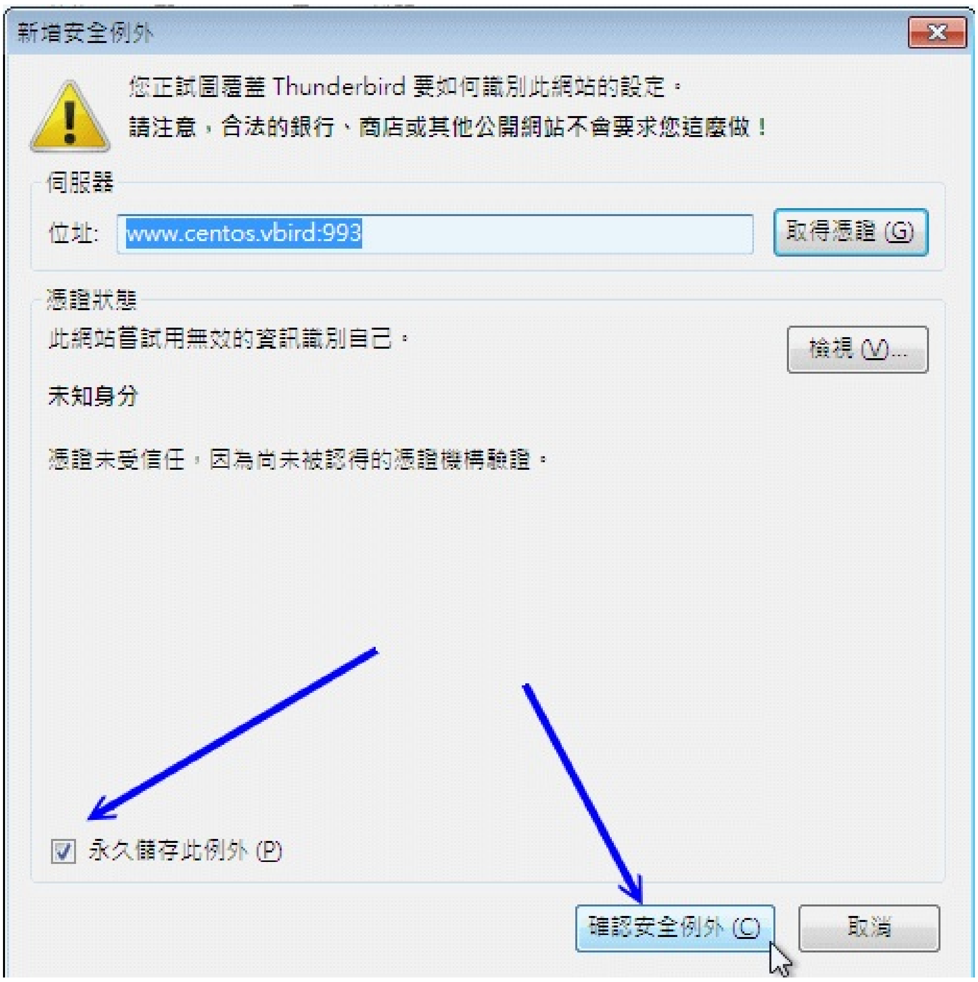


图 22.4-5、取得凭证的示意图

确定凭证 OK、账号密码也 OK 的话，就可以开始使用 Thunderbird 啦！正常使用的图示有点像这样：

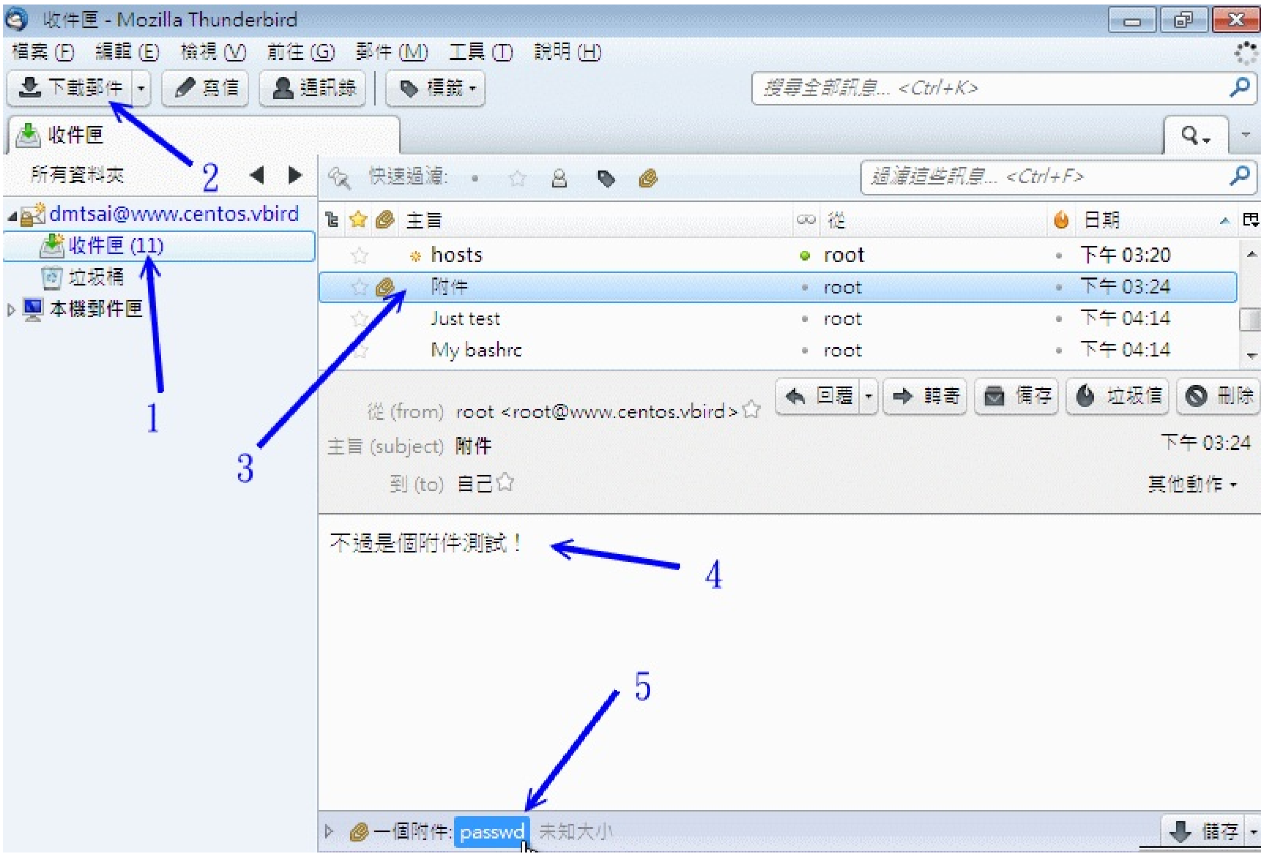


图 22.4-6、Thunderbird 正常操作示意图

如果一切成功顺利，那么你应该会看到如上的画面了！回到刚刚我们查询到的标题名为『附件』的邮件，查阅一下内容，嘿嘿！你会看到内文与附件都是 OK 正常的啦！而且更开心的是，由于是 imaps 的通讯协议，因此 Thunderbird 的内容会与服务器上面的 /var/spool/mail/dmtsai 这个信箱内容同步喔！不像 POP3 抓下来就删除服务器的信件！ 真是好好用的软件啊！ ^_^

Tips: 老实说，由于 gmail 等免费邮件的盛行，目前连 Openwebmail 自由软件都很少人安装了！鸟哥上课时看到的同学， 几乎全部使用 gmail, yahoo mail, hotmail 等的 web 接口的 MUA，根本没人在用本机端的 MUA 了～但是， 某些时刻某些过时的信件还是得要从 web mail 上面捉下来，这时，Thunderbird 就派上用场啰！ ^_^



22.5 邮件服务器的进阶设定

时至今日，邮件攻击主要的问题已经不是病毒与木马了，大多数的垃圾邮件多是钓鱼以及色情广告。 网络钓鱼的问题在于用户的莫名好奇心以及较糟糕的操作习惯，这部份很难处理。 色情广告则是防不胜防，你想出一个过滤机制， 他就使用另一个机制来丢你！用严格的过滤机制吗？又可能将正常的信件抵挡掉，真是要命啊！所以，还是请用户直接删除比较好。 因此，在这一个小节当中，关于收信的过滤机制方面，鸟哥移除了前一版介绍的病毒扫瞄以及自动学习广告机制了。 如果你还是有相关的需要，可能得要自行查查相关的官方网站啰！不好意思啦！

另外，底下主要针对 postfix 的邮件收下过滤处理，以及重新发送的 Relay 过程进行介绍。这两个过程在 postfix 的设定中，主要有几个重要的项目管理：

- smtpd_recipient_restrictions：recipient 是收件者的意思，这个设定值主要在管理『由本机所收下的信件』的功能，因此大部分的设定都是在进行邮件过滤以及是否为可信任邮件的意思。 来源可以是 MTA 或 MUA 的意思；
- smtpd_client_restrictions：client 是客户端的意思，因此主要在管理客户端的来源是否可信任。 可以将非正规的 mail server 来信拒绝掉的！来源当然就是 MUA 啰；
- smtpd_sender_restrictions：sender 是寄件人的意思，可以针对信件来源 (对方邮件服务器) 来进行分析过滤的动作。来源理论上就是 MTA 啦！

22.5.1 邮件过滤一：用 postgrey 进行非正规 mail server 的垃圾信抵挡

早期的广告信很多都是藉由僵尸计算机 (已经被当作跳板但管理员却没有发现或没有处理的主机) 来发送的， 这些僵尸计算机所发送的信件有个很明显的特色，就是『他只会尝试传送该封电子邮件一次， 不论有无成功，该封信就算发出去了，故该信件将被移出队列中。』 不过，合法的 mail server 运作流程就如 22.2.8 分析的一般，在邮件无法顺利寄出时该邮件会暂时放置到队列中一段时间， 并一直尝试将信件寄出的动作，预设直到五天后若还是无法寄出才会将信件退回。

根据这个合法与非法的邮件服务器运作流程而发展出一套所谓的曙光 (postgrey) 软件， 你可以参考底下的几个说明来了解这个软件：

- http://isg.ee.ethz.ch/tools/postgrey/
- http://www.postfix.org/SMTPD_POLICY_README.html

基本上 postgrey 主要的功能是在记录发信来源而已，若发信来源同一封信第一次寄来时，postgrey 默认会抵挡他，并且将来源地址记录起来，在约 5 分钟后，若该信件又传来一次时， 则该信件会被收下来。如此则可以杜绝非发邮件服务器单次发送的问题喔！ ^_^！ 但对于你确定合法的主机则可以开放所谓的『白名单 (whitelist) 』来优先通过而不抵挡。 所以说，他主要是这样进行的：(参考http://projects.puremagic.com/greylisting/whitepaper.html)

1. 确认发信来源是否在白名单中，若是则予以通过；
2. 确认收信者是否在白名单中，若是则予以通过；
3. 确定这封信是否已经被记录起来呢？放行的依据是：
    - 若无此信件的记录，则将发信地址记录起来，并将信件退回；
    - 若有此信件的记录，但是记录的时间尚未超过指定的时间 (预设 5 分钟)，则依旧退回信件；
    - 若有信件的记录，且记录时间已超过指定的时间，则予以通过；

整个过程简单的来说就是这样而已。不过为了要快速的达成 postgrey 的『记录』能力，所以数据库系统又是不可避免的东西。 且 postgrey 是由 perl 写成的，你可能也需要加入很多相依的 perl 模块才行。总的来说，你需要的软件至少要有：

- BerkeleyDB： 包括 db4, db4-utils, db4-devel 等软件：
- Perl： 使用 yum install perl 即可；
- Perl 模块： perl-Net-DNS 是 CentOS 本身有提供的，其他没有提供的可以到 http://rpmfind.net/去搜寻下载。

- 安装流程：

因为 CentOS 官方已经提供了一个连结可以找到所有的在线 yum 安装方式，你可以参考：

- 官网介绍：http://wiki.centos.org/HowTos/postgrey
- 在线安装软件：http://wiki.centos.org/AdditionalResources/Repositories/RPMForge

鸟哥假设你已经下载了 http://packages.sw.be/rpmforge-release/rpmforge-release-0.5.2-2.el6.rf.x86_64.rpm 这个软件且放置到 /root 底下，然后这样做：

```
[root@www ~]# rpm --import http://apt.sw.be/RPM-GPG-KEY.dag.txt
[root@www ~]# rpm -ivh rpmforge-release-0.5.2-2.el6.rf.x86_64.rpm
[root@www ~]# yum install postgrey
```

上述的动作在进行数字签名档案的安装、yum 配置文件的建置，以及最终将 postgrey 透过网络安装起来而已！ 整个流程简单到不行呢！最重要的是，找到适合你的 yum 配置文件软件来安装就是了！

- 启动与设定方式：

因为 postgrey 是额外的一个软件，因此我们还是得要将它视为一个服务来启动，同时 postgrey 是本机的 socket 服务而非网络服务，他只提供给本机的 postfix 来作为一个外挂，因此观察的方式并不是观察 TCP/UDP 之类的联机喔！底下让我们来瞧瞧启动与观察的过程吧！

```
[root@www ~]# /etc/init.d/postgrey start
[root@www ~]# chkconfig postgrey on
[root@www ~]# netstat -anlp &#124; grep postgrey
Active UNIX domain sockets (servers and established)
Proto RefCnt Type    State      PID/Program  Path
unix  2      STREAM  LISTENING  17823/socket /var/spool/postfix/postgrey/socket
```

上表中最重要的就是那个输出的 path 项目啦！/var/spool/postfix/postgrey/socket 是用来做为程序之间的数据交换， 这也是我们的 postfix 要将信件交给 postgrey 处理的一个相当重要的接口！有了这个数据后，接下来我们才能够开始修改 postfix 的 main.cf 啰！

```
[root@www ~]# vim /etc/postfix/main.cf
# 1\. 更改 postfix 的 main.cf 主配置文件资料：
# 一般来说，smtpd_recipient_restrictions 得要手动加入才会更动默认值：
smtpd_recipient_restrictions =
   permit_mynetworks,               &lt;==默认值，允许来自 mynetworks 设定值的来源
   reject_unknown_sender_domain,    &lt;==拒绝不明的来源网域 (限制来源 MTA )
   reject_unknown_recipient_domain, &lt;==拒绝不明的收件者 (限制目标 MTA)
   reject_unauth_destination,       &lt;==默认值，拒绝不信任的目标
   **check_policy_service unix:/var/spool/postfix/postgrey/socket**
# 重点是最后面那一行！就是指定使用 unix socket 来连接到 postgrey 之意。
# 后续我们还有一些广告信的抵挡机制，特别建议您将这个 postgrey 的设定值写在最后，
# 因为他可以算是我们最后一个检验的机制喔！

# 2\. 更改 postgrey 的抵挡秒数，建议将原本的 300 秒 (五分钟) 改为 60 秒较佳：
[root@www ~]# vim /etc/sysconfig/postgrey  &lt;==预设不存在，请手动建立
OPTIONS="--unix=/var/spool/postfix/postgrey/socket --delay=60"
# 重点是 --delay 要抵挡几秒钟，默认值为 300 秒，我们这里改为 60 秒等待。

[root@www ~]# /etc/init.d/postfix restart
[root@www ~]# /etc/init.d/postgrey restart
```

由于过往的经验指出，等待 5 分钟有时候会让某些正常的 mail server 也会被拒绝好久，对于紧急的信件来说，这样有点不妥。 因此，CentOS 官网也建议将这个数值改小一点，例如 60 秒即可。反正，不正常的信件第一次寄就会被拒绝， 等多久似乎也不是这么重要了。然后，在 postfix 的设定中，默认值仅有允许本机设定 (permit_mynetworks) 以及拒绝非信任的目标 (reject_unauth_destination)，鸟哥根据经验，先加入拒绝发件人 (MTA) 的不明网域以及拒绝收件者的不明网域的信件了， 这样也能够减少一堆不明的广告信件。最终才加入 postgrey 的分析。

要注意的是，smtpd_recipient_restrictions 里面的设定是有顺序之分的！以上面的流程来说，只要来自信任用户，该封信件就会被收下会转递，然后不明的来源与目标会被拒绝，不受信任的目标也会被拒绝， 这些流程完毕之后，才开始正常信件的 postgrey 机制处理！这样其实已经可以克服一堆广告信了！ 接下来，让我们测试看看 postgrey 有没有正常运作！请在外部寄一封信到本机来吧！例如寄给 dmtsai@www.centos.vbird， 然后查一下 /var/log/maillog 的内容看看：

```
Aug 10 02:15:44 www postfix/smtpd[18041]: NOQUEUE: reject: RCPT from vbirdwin7[192.168.10
```

鸟哥事先取消 permit_mynetworks 之后才开始测试，测试完毕后又将 permit_mynetworks 加回来才好！这样才能看到上述的资料。 这表示 postgrey 已经开始顺利运作了！并且来源主机的相关记录也已经记载在 /var/spool/postfix/postgrey/ 目录下啰！如此一来您的 postfix 将可以透过 postgrey 来挡掉一些莫名其妙的广告信啰！

- 设定不受管制的白名单：

不过 postgrey 也是有缺点的，怎么说呢？因为 postgrey 预设会先将信件退回去，所以你的信件就可能会发生延迟的问题， 延迟的时间可能是数分钟到数小时，端看你的 MTA 设定而定。如果你想要让『某些信任的邮件主机不需要经过 postgrey 的抵挡机制』时，就得要开放白名单啰！

白名单的开启也很简单啊，直接编写 /etc/postfix/postgrey_whitelist_clients 这个档案即可。假设你要让鸟哥的邮件服务器可以自由的将信寄到你的 MTA 的话，那么你可以在这个档案内加入这一行：

```
[root@www ~]# vim /etc/postfix/postgrey_whitelist_clients
mail.vbird.idv.tw
www.centos.vbird
# 将主机名写进去吧！

[root@www ~]# /etc/init.d/postgrey restart
```

如果你还有更多信任的 MTA 服务器的话，将他写入这个档案当中！那他就可以略过 postgrey 的分析啰！ 更进阶的用法就得要靠您自己去发掘啰！ ^_^

22.5.2 邮件过滤二：关于黑名单的抵挡机制

还记得 22.1.5 讲到的 Open Relay 的问题吧？你的 MTA 可千万不能成为 Open Relay 的状况，否则对你的网络与『信用』影响很大喔！一般来说，只要是 Open Relay 的邮件 MTA 都会被列入黑名单当中， 例如台湾地区的学术网络黑名单以及因特网社会上提供的黑名单数据库：

- http://rs.edu.tw/tanet/spam.html
- http://cbl.abuseat.org/

既然黑名单数据库里面的 mail server 本身就是有问题的邮件主机，那么当黑名单里面的主机想要跟我的 mail server 联机时，我当然可以『合理的怀疑该信件是有问题的！』您说是吧！所以来自黑名单或者是要送至黑名单的信件最好是不要接受啦！

您当然可以自行前往该网站将有问题的主机列表给他加入自己的邮件主机抵挡机制当中， 不过就是不太人性化！既然因特网社会已经提供了黑名单数据库了，我们就可以利用这个数据库来抵挡嘛！ 在决定是否进行 Relay 之前，先要求我们的 postfix 前往追踪黑名单的数据库，若目标的 IP 或主机名是黑名单的一员，则我们就将该信件拒绝啰！

Postfix 设定黑名单检验眞的很简单，你只要这样做即可：

```
[root@www ~]# vim /etc/postfix/main.cf
smtpd_recipient_restrictions =
   permit_mynetworks,
   reject_unknown_sender_domain,
   reject_unknown_recipient_domain,
   reject_unauth_destination,
 reject_rbl_client cbl.abuseat.org,
   reject_rbl_client bl.spamcop.net,
   reject_rbl_client cblless.anti-spam.org.cn,
   reject_rbl_client sbl-xbl.spamhaus.org,
   check_policy_service unix:/var/spool/postfix/postgrey/socket
# 请注意整个设定值的顺序才好！在 postgrey 之前先检查是否为黑名单！

smtpd_client_restrictions =
     check_client_access hash:/etc/postfix/access,
     reject_rbl_client cbl.abuseat.org,
     reject_rbl_client bl.spamcop.net,
     reject_rbl_client cblless.anti-spam.org.cn,
     reject_rbl_client sbl-xbl.spamhaus.org
# 这个设定项目则是与客户端有关的设定！拒绝客户端本身就是黑名单的一员！

smtpd_sender_restrictions = reject_non_fqdn_sender,
   reject_unknown_sender_domain
# 此项目则在抵挡不明的送件者主机网域啰！与 DNS 有关系的哪！

[root@www ~]# /etc/init.d/postfix restart
```

上表当中的特殊字体部分『reject_rbl_client』是 postfix 内的一个设定项目，后面可以接因特网上提供的黑名单！ 您得要注意的是，这个黑名单数据库可能会持续的变动，请您先以 dig 的方式检查每个数据库是否眞的存在， 如果存在才加以设定在您的主机上头啊！(因为因特网上头很多文献所提供的黑名单数据库似乎已经不再持续服务的样子！)

- 检查你的邮件服务器是否在黑名单当中？

既然黑名单数据库所记录的是不受欢迎的来源与目标 MTA ，那么您的 MTA 当然最好不要在该数据库中嘛！ 同时这些数据库通常也都有提供检测的功能，所以你也可以用该功能来检查你的主机是否『记录有案』呢？ 你可以这样处理的：

1. 是否已在黑名单数据库中： 确认的方法很简单，直接到『http://cbl.abuseat.org/lookup.cgi 』输入您的主机名或者是 IP ，就可以检查是否已经在黑名单当中；

2. 是否具有 Open Relay： 如果要测试你的主机有没有 Open Relay ，直接到『http://rs.edu.tw/tanet/spam.html 』这个网页， 在这个网页的最下方可以输入你的 IP 来检查，注意喔，不要使用别人的 email IP 呐！ 此时该主机会发出一封 mail 的测试信看看你的 mail server 会不会主动的代转， 然后将结果回报给您。要注意的是，回传的网页可能有编码的问题，如果出现乱码时，请调整为 big5 编码即可。

3. 如何移除： 如果被检查出，您的主机已经在黑名单当中，那么请立刻将 Open Relay 的功能关闭，改善你的 Mail Server 之后，你可能还要到各个主要的 Open Relay 网站进行移除的工作。如果是学术网络的话， 请与您单位的管理员联络。至于一般常见的黑名单数据库则通常会主动的帮您移除，只不过需要一些时间的测试就是了。

总之您必须要确定你不在黑名单当中，且最好将黑名单的来源给拒绝掉！搞定！ ^_^

22.5.3 邮件过滤三：基础的邮件过滤机制

在整封信的传送流程当中，客户端若通过主机的重重限制后，最终应该可以到达邮件队列当中。 而由队列当中要送出去或者是直接送到 mailbox 就得要透过 MDA 的处理。MDA 可以加挂很多机制呢！ 尤其是他可以过滤某些特殊字眼的广告信件或病毒信件呢！ MDA 可以透过分析整封信件的内容 (包括标头以及内文) 来撷取有问题的关键词，然后决定这封信的『命运』说！

咱们的 postfix 已经有内建可以分析标头或者是内文的过滤机制了，那就是 /etc/postfix/ 目录下的 header_checks 以及 body_checks 这两个档案啊！在预设的情况下这两个档案不会被 postfix 使用， 你必需要用底下的设定来启用他：

```
[root@www ~]# vim /etc/postfix/main.cf
header_checks = regexp:/etc/postfix/header_checks
body_checks = regexp:/etc/postfix/body_checks
# 那个 regexp 代表的是『使用正规表示法』的意思啦！

[root@www ~]# touch /etc/postfix/header_checks
[root@www ~]# touch /etc/postfix/body_checks
[root@www ~]# /etc/init.d/postfix restart
```

接下来你必需要自行处理 header_checks 以及 body_checks 的规则设定，在设定前请您确认『 你对于正规表示法是熟悉的 』才行！因为很多信息都必需要透过正规表示法来处理啦！然后开始设定的依据是：

- 只要是 # 代表该行为批注，系统或直接略过；
- 在默认的规则当中，大小写是视为相同的；
- 规则的设定方法为：

  > `> /规则/ 动作 显示在登录文件里面的讯息 >`

  请注意，要使用两个斜线『 / 』将规则包起来喔！举个例子来说明：例如我想要 (1)抵挡掉标题为 A funny game 的信件，(2)并且在登录文件里面显示 drop header deny，则可以在 header_chekcs 档案中可以这样写：

  > `> /^Subject:.*A funny game/ DISCARD drop header deny >`

- 关于动作有底下几个动作：

  - REJECT ：将该封信件退回给原发信者；
  - WARN ：将信件收下来，但是将该封信的基本数据记录在登录文件内；
  - DISCARD：将该封信件丢弃，并不给予原发信者回应！

鸟哥自己有作一些规则的比对，只不过.....效能不好！如果您有兴趣的话，可以自行下载来看看， 不过，使用的后果请自行评估！因为每个人的环境都不一样嘛！

- header: http://linux.vbird.org/linux_server/0380mail/header_checks
- body: http://linux.vbird.org/linux_server/0380mail/body_checks

记得，如果你自行修改过这两个档案后，务必要检查一下语法才行！

```
[root@www ~]# postmap -q - regexp:/etc/postfix/body_checks \
&gt;  &lt; /etc/postfix/body_checks
```

如果没有出现任何错误，那就表示您的设定值应该没有问题啦！另外，你也可以使用 procmail 这个抵挡的小程序来处理。 不过，鸟哥觉得 procmail 在大型邮件主机当中，分析的过程太过于繁杂，会消耗很多 CPU 资源，因此后来都没有使用这玩意儿了。

22.5.4 非信任来源的 Relay：开放 SMTP 身份认证

在图 22.1-1 的流程当中，由 MUA 透过 MTA 来寄发信件时 (具有 Relay 的动作时)，理论上 MTA 必需要开放信任用户来源才行，这就是为啥我们必需要在 main.cf 里头设定 smtpd_recipient_restrictions 那个设定项目的原因了 (mynetworks)！不过人总有不方便的时候，举例来说，如果你的客户端使用的是拨接制的 ADSL 所以每次取得的 IP 都非固定，那如何让你的用户使用你的 MTA ？很麻烦是吧？这个时候 SMTP 认证或许有点帮助。

什么是 SMTP 呢？就是让你在想要使用 MTA 的 port 25 (SMTP 协议) 时，得要输入账号密码才能够使用的意思！既然有了这个认证的功能，于是乎，你就可以不用设定 MTA 的信任用户项目！举例来说，在本章提到的环境下，你可以不用设定 mynetworks 这个设定值啊！启动 SMTP 认证，让你的用户需要输入账密才能 Relay 啰！那如何让 SMTP 支持身份认证？咱们的 CentOS 已经有提供内建的认证模块，那就是 Cyrus SASL 这个软件的帮忙啦！

Cyrus SASL (http://cyrusimap.web.cmu.edu/) 是 Cyrus Simple Authentication and Security Layer 的缩写，他是一个辅助的软件。在 SMTP 认证方面，Cyrus 主要提供了 saslauthd 这个服务来进行账号密码的比对动作！也就是说：当有任何人想要进行邮件转递功能时， Postfix 会联络 saslauthd 请其代为检查账号密码，若比对通过则允许客户端开始转寄信件。

好了，如果你想要使用最简单的方式，就是直接透过 Linux 自己的帐密来进行 SMTP 认证功能，而不使用其他如 SQL 数据库的身份认证时，在 CentOS 当中你应该要这样做：

1. 安装 cyrus-sasl, cyrus-sasl-plain, cyrus-sasl-md5 等软件；
2. 启动 saslauthd 这个服务；

3. 设定 main.cf 让 postfix 可以与 saslauthd 联系；
4. 客户端必需要在寄信时设定『邮件主机认证』功能。

如此一来客户端才能够启动 SMTP AUTH 喔！关于软件安装方面，请使用 yum 直接安装吧！不再多啰唆！底下我们由启动 saslauthd 这个服务开始谈起吧！

- 启动 saslauthd 服务：进行 SMTP 明码身份验证功能

saslauthd 是 Cyrus-SASL 提供的一个账号密码管理机制，他能够进行挺多的数据库验证功能， 不过这里我们仅使用最单纯简单的明码验证 (PLAIN)！如果我们想要直接使用 Linux 系统上面的用户信息， 也就是 /etc/passwd, /etc/shadow 所记载的账号密码相关信息时，可以使用 saslauthd 提供的『 shadow 』这个机制， 当然也能使用『 pam 』啦！更多的 saslauthd 联机至 MTA 的机制请『 man saslauthd 』来查阅吧。 由于我们的帐密可能来自网络其他类似 NIS 服务器，因此这里建议可以使用 pam 模块喔！

saslauthd 的启动眞是好简单，首先你必需要选择密码管理机制，这个可以使用底下的方式处理：

```
# 1\. 先了解你的 saslauthd 有支持哪些密码管理机制：
[root@www ~]# saslauthd -v
saslauthd 2.1.23
authentication mechanisms: getpwent kerberos5 &lt;u&gt;pam&lt;/u&gt; rimap &lt;u&gt;shado
# 上列的特殊字体部分就是有支持的！我们要直接用 Linux 本机的用户信息，
# 所以用 pam 即可，当然也能够使用 shadow 啦。

# 2\. 在 saslauthd 配置文件中，选定 pam 的验证机制：
[root@www ~]# vim /etc/sysconfig/saslauthd
MECH=pam  &lt;==其实这也是默认值啊！
# 这也是默认值，有的朋友喜欢单纯的 shadow 机制，也可以啦！

# 3\. 那就启动吧！
[root@www ~]# /etc/init.d/saslauthd start
[root@www ~]# chkconfig saslauthd on
```

之后我们必需要告知 Cyrus 这个咚咚使用来提供 SMTP 服务的程序为 saslauthd 才行，设定的方法很简单：

```
[root@www ~]# vim /etc/sasl2/smtpd.conf
log_level: 3                &lt;==登录文件信息等级的设定，设定 3 即可
pwcheck_method: saslauthd   &lt;==就是选择什么服务来负责密码的比对啊
mech_list: plain login      &lt;==那么支持的机制有哪些之意！
```

我们可以使用 mech*list* 列出特定支持的机制。而且 *saslauthd* 是个很简单的账号密码管理服务，你几乎不需要进行什么额外的设定，直接启动他就生效了！眞是好方便！ ^^

- 更改 main.cf 的设定项目：让 postfix 支持 SMTP 身份验证

那我们的 postfix 该如何处理呢？其实设定眞的很简单，只要这样做就好了：

```
[root@www ~]# vim /etc/postfix/main.cf
# 在本档案最后面增加这些与 SASL 有关的设定资料：
smtpd_sasl_auth_enable = yes
smtpd_sasl_security_options = noanonymous
broken_sasl_auth_clients = yes
# 然后找到跟 relay 有关的设定项目，增加一段允许 SMTP 认证的字样：
smtpd_recipient_restrictions =
     permit_mynetworks,
     permit_sasl_authenticated,  &lt;==重点在这里！注意顺序！
     reject_unknown_sender_domain,
     reject_unknown_recipient_domain,
     reject_unauth_destination,
     reject_rbl_client cbl.abuseat.org,
     reject_rbl_client bl.spamcop.net,
     reject_rbl_client cblless.anti-spam.org.cn,
     reject_rbl_client sbl-xbl.spamhaus.org,
     check_policy_service unix:/var/spool/postfix/postgrey/socket

[root@www ~]# /etc/init.d/postfix restart
```

上面关于 SASL 的各个项目的意义是这样的：

- smtpd_sasl_auth_enable 就是设定是否要启动 sasl 认证的意思，如果设定启动后 postfix 会主动去加载 cyrus sasl 的函式库， 而该函式库会依据 /etc/sasl2/smtpd.conf 的设定来连结到正确的管理账号与密码的服务。

- smtpd_sasl_security_options 由于不想要让匿名者可以登入使用 SMTP 的 Relay 功能，于是这个项目中只要设定 noanonymous 即可。

- broken_sasl_auth_clients 这个是针对早期非正规 MUA 的设定项目，因为早期软件开发商在开发 MUA 时没有参考通讯协议标准， 所以造成在 SMTP 认证时可能会发生的一些困扰。这些有问题的 MUA 例如 MS 的 outlook express 第四版就是这样！ 后来的版本应该没有这个问题。所以这个设定值你也可以不要设定！

- smtpd_recipient_restrictions 最重要的就是这里啦！我们的 sasl 认证可以放在第二行，在局域网络这个可信任区域的后面加以认证。 上表的设定意义是：局域网络内的 MUA 不需要认证也能够进行 relay ，而非区网内的其他来源才需要进行 SMTP 认证之意。

设定完毕也重新启动 postfix 之后，我们先来测试看看是否真的提供认证了？

```
[root@www ~]# telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.
220 www.centos.vbird ESMTP Postfix
ehlo localhost
250-www.centos.vbird
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-AUTH LOGIN PLAIN    &lt;==你得要看到这两行才行呦！
250-AUTH=LOGIN PLAIN
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN
quit
221 2.0.0 Bye
```

- 在客户端启动支持 SMTP 身份验证的功能：以 thunderbird 设定为例

既然已经在 MTA 设定了 SMTP 身份验证，那么我们 MUA 当然要传送账号、密码给 MTA 才能通过 SMTP 的验证嘛！ 所以，在 MUA 上面就得要加上一些额外的设定才行。我们依旧以 Thunderbird 来作为介绍，请打开 thunderbird，选择『工具』--> 『账号设定』后会出现如下画面：

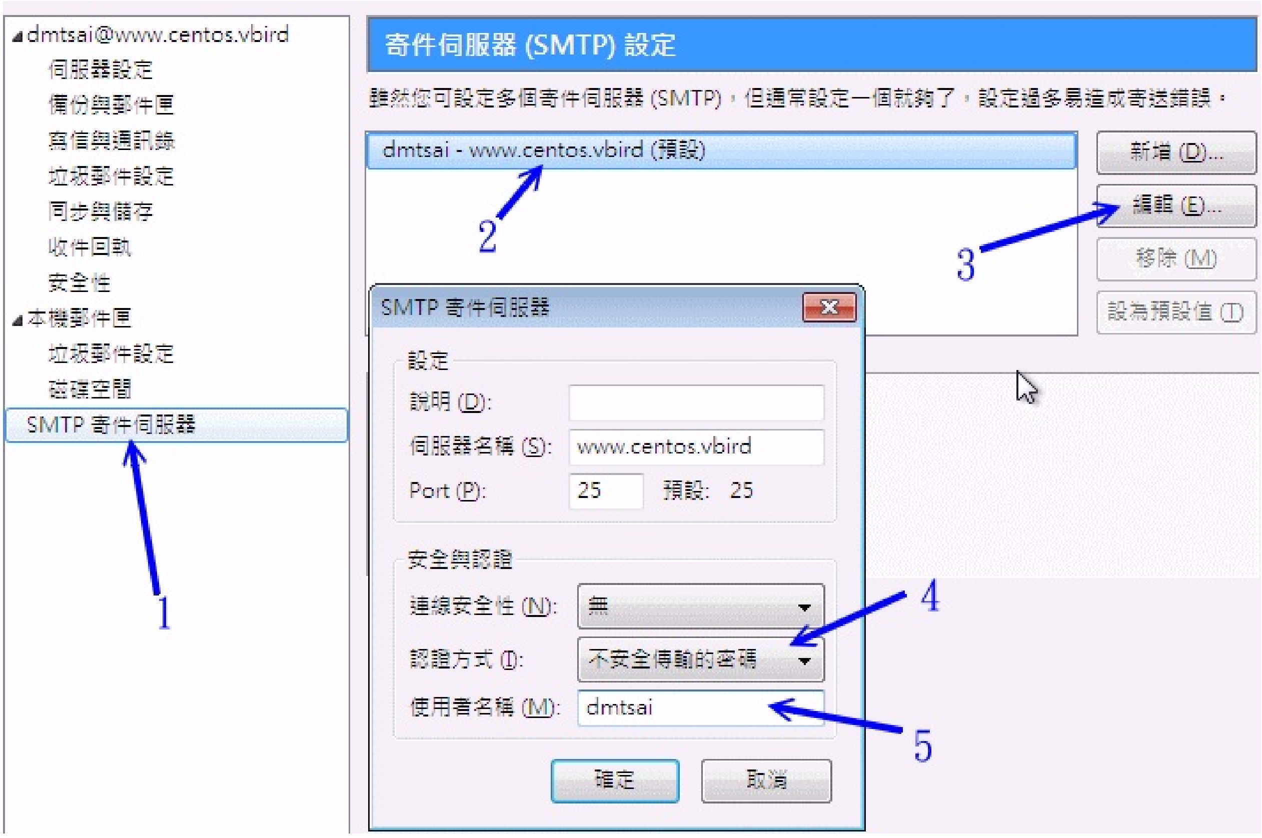


图 22.5-1、在 Thunderbird 软件中设定支持 SMTP 验证的方式

请依据上图的箭头号码来指定，先选择 (1)SMTP 寄件服务器；，然后选择所需要的寄件 SMTP 服务器后，点选 (3)编辑， 就会出现上图中的窗口项目 。选择 (4)不安全传输的密码后，在 (5)填入你要使用的账号即可。 如果要测试的话，记得此客户端不要在局域网络内，否

则将不会经过认证的阶段，因为我们的设定以信任网域为优先嘛！

如果一切都顺利的话，那么当客户端以 SMTP 来验证时，你的登录档应该会出现类似底下的讯息才是：

```
[root@www ~]# tail -n 100 /var/log/maillog &#124; grep PLAIN
Aug 10 02:37:37 www postfix/smtpd[18655]: 01CD43712: client=vbirdwin7
[192.168.100.30], sasl_method=PLAIN, sasl_username=dmtsai
```

22.5.5 非固定 IP 邮件服务器的春天： relayhost

我们上面提到，如果你要架设一部合法的 MTA 最好还是得要申请固定的 IP 以及正确对应的反解比较恰当。 但如果你一定要用浮动 IP 来架设你的 MTA 的话，也不是不可以啦，尤其今年 (2011) 光纤到府已经可达 50M/5Mbps 的下载/上传速度了！你当然可以用家庭网络来架站啊！只不过你就得要透过上层 ISP 所提供的 relay 权限啰！这是怎么回事啊？让我们来看看一个实际的案例：

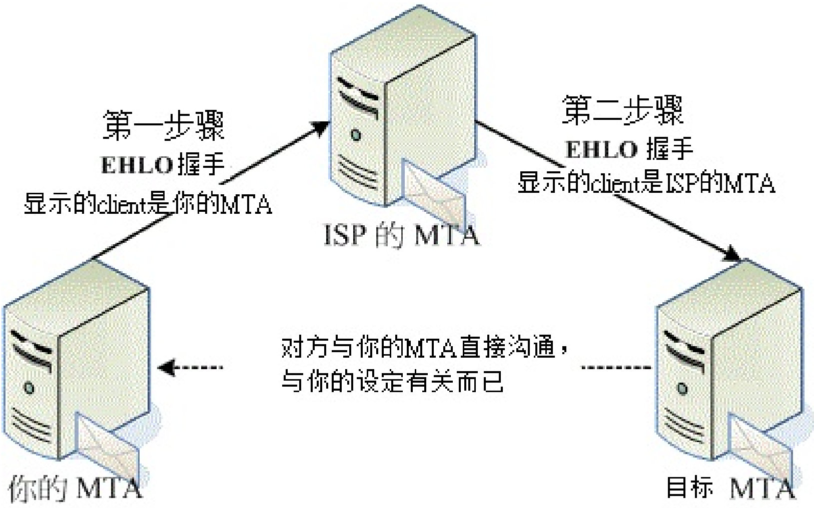


图 22.5-2、Relayhost：利用 ISP 的 MTA 进行邮件转递

当你的 MTA 要传信件给目标 MTA 时，如果直接传给目标 MTA，由于你的 IP 可能是非固定的，因此对方 MTA 恐怕会把你当成是垃圾来源！那如果我们可以透过 ISP 进行转递呢？从上面的图示来看，当你要传给目标 MTA 时： (1)先将信件交给妳的 ISP，因为你是 ISP 的客户，通常来信都会被 ISP 接受，因此这个时候这封信就会被你的 ISP 给 relay 出去； (2)被 ISP 所 relay 的信件到目标 MTA 时，对方会判断是来自那部 ISP 的 MTA，当然是合法的 mail server， 所以该封信件就毫无疑问的被收下啰！ ^_^

不过想要以此架构来架设你的 MTA 仍有许多需要注意的地方：

- 你还是得要有一个合法的主机名，若要省钱，可以使用 DDNS 来处理；
- 你上层的 ISP 所提供的 MTA 必需要有提供你所在 IP 的 relay 权限；
- 你不能使用自定义的内部 DNS 架构了，因为所有 relay 的信都会被送至 ISP 的 MTA

尤其是最后一点，因为所有外送的信件全部都会被送到 ISP 处，所以像我们之前自己玩的 centos.vbird 这种非合法的领域数据就没用了！为什么呢？你想想看，如果你要将信件送给 www.centos.vbird， 但由于上述 relayhost 的功能，所以这封信会被传到 ISP 的 MTA 来处理，但 ISP 的 MTA 会不会认识你的 centos.vbird？这样说，可以理解了吧？

说是挺难的，做起来却很简单，只要在 main.cf 里面加设一段数据即可。 假设你的环境是台湾地区的 hinet 所提供的用户，而 hinet 提供的邮件主机为 ms1.hinet.net ， 则你可以直接这样设定：

```
[root@www ~]# vim /etc/postfix/main.cf
# 加入底下这一行就对啦！注意那个中括号！
relayhost = [ms1.hinet.net]

[root@www ~]# /etc/init.d/postfix restart
```

之后你只要尝试寄一封信出去看看，就会了解这封信是如何寄送的了。看一下登录档的内容会像这样：

```
[root@www ~]# tail -n 20 /var/log/maillog
Aug 10 02:41:01 www postfix/smtp[18775]: AFCA53713: to=&lt;qdd@mail.ksu.edu.tw&gt;,
relay=ms1.hinet.net[168.95.4.10]:25, delay=0.34, delays=0.19/0.09/0.03/0.03,
dsn=2.0.0, status=sent (250 2.0.0 Ok: queued as F0528233811)
```

是吧！经由上层 ISP 来转寄啦！如此一来，你的 MTA 感觉上就似乎是部合法的 MTA 啰！ 不过，可别利用这个权限来滥发广告信啊！因为您所透过的那个 ISP 邮件主机可是有记录你的 IP 来源， 如果你乱来的话，后果可是不堪设想喔！切记切记！

22.5.6 其他设定小技巧

除了之前谈到的几个主要的设定之外， postfix 还有提供一些不错的设定要给大家使用的喔！我们可以一个一个来来看看：

- 单封信件与单个邮件信箱的大小限制

在预设的情况下，postfix 可接受的单封信件最大容量为 10MBytes ，不过这个数值我们是可以更改的， 动作很简单：

```
[root@www ~]# vim /etc/postfix/main.cf
message_size_limit =   40000000
[root@www ~]# postfix reload
```

上面的单位是 bytes，所以我将单封信件可接受大小改为 40MByte 的意思啦！请按照你的环境来规定这个数值。 而从前我们要管制 /var/spool/mail/account 大多是使用文件系统内的 quota 来达成， 现在的 postfix 不需要啦！可以这样做：

```
[root@www ~]# vim /etc/postfix/main.cf
mailbox_size_limit = 1000000000
[root@www ~]# postfix reload
```

我给每个人 1GB 的空间啊！^_^

- 寄件备份：SMTP 自动转寄一份到备份匣

收件备份我们知道可以使用 /etc/aliases 来处理的，但是如果想要送件也备份呢？利用底下的方式即可：

```
[root@www ~]# vim /etc/postfix/main.cf
always_bcc = some@host.name
[root@www ~]# postfix reload
```

如此一来任何人寄出的信件都会复制一份给 some@host.name 那个信箱。不过，除非您的公司很重视一些商业机密， 并且已经公告过所有同仁，否则进行这个设定值，鸟哥个人认为侵犯隐私权很严重！

- 配置文件的权限问题：权限错误会不能启动 postfix

这部份我们以 Sendmail 官方网站的建议来说明喔！其实也适用于 postfix 的啦！ 其中，大部分是在于『目录与档案权限』的设定要求上面：

- 请确定 /etc/aliases 这个档案的权限，仅能由系统信任的账号来修改，通常其权限为 644 ；
- 请确定 Mail server 读取的数据库 (多半在 /etc/mail/ 或 /etc/postfix/ 底下的 *.db 档案)，例如 mailertable, access, virtusertable 等等，仅能由系统信任的用户读取，其他一概不能读取，通常权限为 640 ；
- 系统的队列目录 (/var/spool/mqueue 或 /var/spool/postfix) 仅允许系统读取，通常权限为 700 ；
- 请确定 ~/.forward 这个档案的权限也不能设定成为任何人均可查阅的权限，否则您的 e-mail 数据可能会被窃取～
- 总之，一般用户能够不用 ~/.forward 与 aliases 的功能，就不要使用！

不过整体的使用上还是需要身为网站管理员的您多费心！多多观察登录档啊！

- 备份资料：与 mail 有关的目录是哪些？

不管什么时候，备份总是重要的！那么如果我是单纯的 Mail Server 而已，我需要的备份数据有哪些呢？

- /etc/passwd, /etc/shadow, /etc/group 等与账号有关的资料；
- /etc/mail, /etc/postfix/ 底下的所有档案数据；
- /etc/aliases 等等 MTA 相关档案；
- /home 底下的所有用户数据；

- /var/spool/mail 底下的档案与 /var/spool/postfix 邮件队列档案；
- 其他如广告软件、病毒扫瞄软件等等的设定与定义档。

- 错误检查：查出不能启动 postfix 的问题流程

虽然 Mail 很方便，但是仍然会有无法将信件寄出的时候！如果您已经设定好 MTA 了，但是总是无法将邮件寄出去，那可能是什么问题呢？你可以这样追踪看看：

1. 关于硬件配备： 例如，是否没有驱动网卡？是否调制解调器出问题？是否 hub 热当啦？是否路由器停止服务等等的！

2. 关于网络参数的问题： 如果连不上 Internet ，那么哪里来的 Mail Server 呢？所以请先确认你的网络已经正常的启用了！ 关于网络的确认问题，请查阅第六章网络侦错来处理。

3. 关于服务的问题： 请务必确认与 mail server 有关的埠口已经顺利启动！例如 port 25, 110, 143, 993, 995 等等，使用 netstat 指令即可了解是否已经启动该服务！

4. 关于防火墙的问题： 很多时候，很多朋友使用 Red Hat 或者其他 Linux distribution 提供的防火墙设定软件，结果忘了启动 port 25 与 port 110 的设定，导致无法收发信件！请特别留意这个问题喔！可以使用 iptables 来检查是否已经启用该 port 呢！其余请参考第九章防火墙设定喔！

5. 关于配置文件的问题： 在启动 postfix 或者是 sendmail 之后，在登录档当中仔细看看有无错误讯息发生？ 通常如果设定数据不对，在登录文件当中都会有记载错误的地方。

6. 其他档案的设定问题： (1)如果发现只有某个 domain 可以收信，其他的同一主机的 domain 无法收信，需要检查 $mydestination 的设定值才行； (2)如果发现邮件被挡下来了！而且老是显示 reject 的字样，那么可能被 access 挡住了； (3)如果发现邮件队列 (mailq) 存在很多的邮件，可能是 DNS 死掉了，请检查 /etc/resolv.conf 的设定是否正确！

7. 其他可能的问题： 最常发生的就是认证的问题了！这是由于使用者没有在 MUA 上面设定『我的邮件需要认证』的选项啦！ 请叫你的用户赶紧勾选吧！

8. 还是不知道问题的解决方案： 如果还是查不出问题的话，那么请务必检查您的 /var/log/maillog (有的时候是 /var/log/mail ，这个要看 /etc/syslog.conf 的设定)，当你寄出一封信的时候，例如 dmtsai 寄给 bird2@www.centos.vbird 时，那么 maillog 档案里面会显示出两行，一行为 from dmtsai 一行为 to bird2@www.centos.vbird， 也就是『我由哪里收到信，而这封信会寄到哪里去！』的意思，由这两行就可以了解问题了！尤其是 to 的那一行，里面包含了相当多的有用信息，包括邮件无法传送的错误原因的纪录！ 如果您对于登录档不熟，请拿出『基础学习篇』里面的『第十九章、认识登录档』一文吧！

22.6 重点回顾

- 电子邮件服务器的设定需要特别留意，以免被作为广告信与垃圾信的跳板；
- Mail server 使用的主机名至少需要 A 的 DNS 标志，不过最好能够具有 MX 标志为宜，且正反解最好成对， 比较可以避免大型 mail server 的抵挡；
- 邮件服务器主要是指 SMTP (简单邮件传送协议) 而已，不过要架设一部可利用类似 Thunderbird 收发的邮件服务器， 最好能够具有 SMTP 以及 POP3 等通讯协议；
- 电子邮件传送的组件，主要有 MUA, MTA, MDA 以及最终的 Mailbox 等等；
- 电子邮件服务器最需要搞定的地方其实是 Relay 的功能，千万不可 Open Relay 喔！
- 一封电子邮件至少含有 header 以及 body 等数据在内；
- 常见的可以启动 SMTP 的软件有 sendmail, postfix 及 qmail 等等。
- 为避免收到大量的广告信，建议您不要将 email address 放在因特网上，若需要某些功能必需将邮件地址放在网络上时， 最好能够拥有两个邮件地址，一个用来公开，一个则用来作为自己的主要联络之用。

22.7 本章习题

- 当你利用你的 MTA 发信时，结果竟然被退信，退信的讯息 (/var/log/maillog) 最主要的错误是『mail loop to me』， 请问可能的发生原因及处理方式为何？可能发生的原因是由于你的 MTA 设定项目方面的主机名错误。判断你的 MTA 主机有多个 IP 存在， 不过你并未完全写入配置文件中，因此造成某些主机名无法被 MTA 所接收之故。在 sendmail 方面， 你只要将需要的主机名写入 /etc/mail/local-host-names 即可，如果是 postfix ，则在 /etc/postfix/main.cf 当中修改 $mydestination 那个设定项目即可。
- 请列出四个 Mail Server 的相关的组件，以及其功用为何？
    - Mail Client ：邮件客户端，其实就是使用 mail 的那位用户所在的计算机即可称为 mail client ；
    - Mail User Agent ：为一个应用软件，主要的功能就是收受邮件主机的电子邮件，以及提供用户浏览与编写邮件的功能；
    - Mail Transfer Agent ：为在计算机与本地端 Mail server 或 Internet 上面的 Mail server 传送讯息与邮件的主机；
    - Mail Dilivery Agent ：主要的功能就是将 MTA 所收受的本机信件，放置到本机账户下的邮件档案中 (Mailbox)！
- POP3 与 SMTP 的功能为何？
    - SMTP 为使用于 MUA 或 MTA 与 MTA 之间的传输协议，通常使用 port 25 ，只要主机支持 SMTP ，并且其他 relay 的条件能配合，就可以进行邮件传递！
    - POP3 可以提供使用者经由 MUA 到 MTA 下载邮件，同时并可将邮件从主机上面删除！
- 请简单的说明 DNS 里面 MX 标志与 Mail 的关系为何？MX recode 可以可以让 mail server 经由 MX 以及 A (address) 这个记录来进行 mail gateway 与 mail route 的功能！能够达到的作用相当的多！

- 什么是 mailling list ？在 postfix 底下有什么方法可以不藉由其他的软件达到 mailling list 的功能？Mailling list 就是将使用者寄给一个账号邮件时，该账号会主动的将该邮件传送到所有的用户去！有点类似目前的电子报！在 sendmail 底下，我们可以透过 aliases (需配合 newaliases) 以及 ~/.forward 来达成喔！
- 如何察看邮件队列的内容，以及邮件队列内容放置在何方？使用 mailq 即可知道目前邮件队列的内容，而邮件队列虽然可以透过 sendmail.cf 来修改，不过，预设情况下，都是以 /var/spool/mqueue 为邮件队列目录。
- 什么是 Open Relay？所谓的 Open Relay 就是，不论发信端来自何处，您的 Open Relay 的主机均可以帮发信端将信件发送出去，这个称为 Open Relay 。如果您的 mail server 具有 open relay 的情况，那么很容易遭受到垃圾邮件的填充， 不但造成网络带宽的耗损，也容易让您的主机被列入黑名单当中！

- 如果要让 Postfix 可以收发来自非本机的外部信件，您可以修改 main.cf 里面的什么参数？需要在 main.cf 里面修改的变量主要有：

  1. 当 Client 来自信任的网域，也就是 IP 符合 $mynetworks 的设定值时；
  2. 当 Client 来自信任的机器，也就是主机名符合 $relay_domains 的设定项目时；
  3. 当 Client 来自不信任的网域，但是去的目的地主机端符合 $relay_domains 的设定时。

- 如何察看您目前的 Postfix 服务器的所有设定参数？(使用什么指令？)利用 postconf -n 可以察看『目前 main.cf 里面设定的参数』，而如果要看所有的参数，则直接使用 postconf 即可！

- Mail Server 能否运作与 DNS (MX 与 A recode)的相关性为何？目前因特网社会合法的 Mail server 通常仅会针对具有 MX 标志的邮件主机发出信件而已。 而如果有多重 MX 时，首先会选择最小 MX 主机寄信，依序处理。而最终依据 MX 主机的 A 标志来查得最终目标。
- 什么是 smtp, pop3 以及 imap 协议，他们的用途分别是什么？

  - smtp：用来传递邮件的协议，通常我们称为 MTA 即是此一协议所达成
  - pop3：让 client 端向主机端要求收信的协议，通常预设收信完成后，主机端的 mail box 会被删除；
  - imap：与 pop3 类似，不过 imap 允许用户在主机的家目录建立邮件数据匣

22.8 参考数据与延伸阅读

- Sendmail 官方网站： http://www.sendmail.org
- Postfix 官方网站： http://www.postfix.org
- Cyrus-SASL 官方网站： http://asg.web.cmu.edu/cyrus/download/sasl/doc/
- Procmail 官方网站： http://www.procmail.org
- Open Relay Database： http://www.ordb.org/ (很可惜，已于 2006/12/18 关站)
- Study Area 之邮件架设：http://www.study-area.org/linux/servers/linux_mail.htm
- SMTP 认证系统的建置：http://beta.wsl.sinica.edu.tw/~ylchang/Email/sendmail-auth/
- 台湾学术网络黑名单网页： http://rs.edu.tw/tanet/spam.html
- 卧龙小三的 Procmailrc 范例：ftp://ftp.tnc.edu.tw/pub/Sysop/MAIL/procmailrc
- 林克敏主任文件集之 Procmail 范例：http://freebsd.lab.mlc.edu.tw/procmail.htm
- Postgrey 官方网站：http://isg.ee.ethz.ch/tools/postgrey/
- Postfix 针对 Postgrey 的设定： http://www.postfix.org/SMTPD_POLICY_README.html
- 一些 postfix 的 relay 机制设定： http://jimsun.linxnet.com/misc/postfix-anti-UCE.txt
- 小州的 postfix 设定： http://phorum.study-area.org/viewtopic.php?t=30716
- POSTFIX 技术手札，Ralf Hildebrandt/Patrick Koetter 合着，上奇出版，2005年。
- TWU2 兄在酷学园所发表的自制邮件过滤软件： http://phorum.study-area.org/viewtopic.php?t=38649
- Amavis-new 一个在 MTA 与队列间的服务：http://www.ijs.si/software/amavisd/
- 广告信件分析软件：http://spamassassin.apache.org/index.html
- Steven 的垃圾信抵挡机制：http://www.l-penguin.idv.tw/article/postfix_spam-spamassassin.htm

---

2006/11/13：准备将原本的 sendmail 以及 postfix 整合成为一篇专门介绍 Mail server 的小文章啰！ 2006/11/14：原本的 sendmail 请参考这里，原本的 Postfix 则请参考这里。至于人数统计则以 sendmail 原本网页增加。 2006/11/30：加上了邮件扫瞄与广告信抵挡的 spamassassin 机制，呼呼！好累～ 2006/12/05：加上自动学习广告信抵挡机制方面的简单介绍。 2007/02/07：新增不要丢弃 exe 附文件名的信息！ 2007/02/27：感谢网友 Cheng-Lin Yang 提供的意见，在黑名单数据库增加了 http://www.anti-spam.org.cn/, http://www.spamhaus.org/ 2007/04/05：感谢 chunkit 兄的来信告知，将原本的 mail localhost 25 改成 telnet localhost 25 ！ 2010/07/20：感谢 Patrick 的告知，学术网络的 spam 网址应为 http://rs.edu.tw/tanet/spam.html 2011/06/05：将旧的基于 CentOS4.x 的版本移动到此处 2011/06/13：不愿意再讲 telnet 的 mail 功能，加入的是 mutt 这个可以联网的好物！ 2011/07/07：这个月实在很忙碌，所以短短一篇 mail server 改了一整个月... 2011/08/10：将基于 CentOS 5.x 的版本移动到此处
